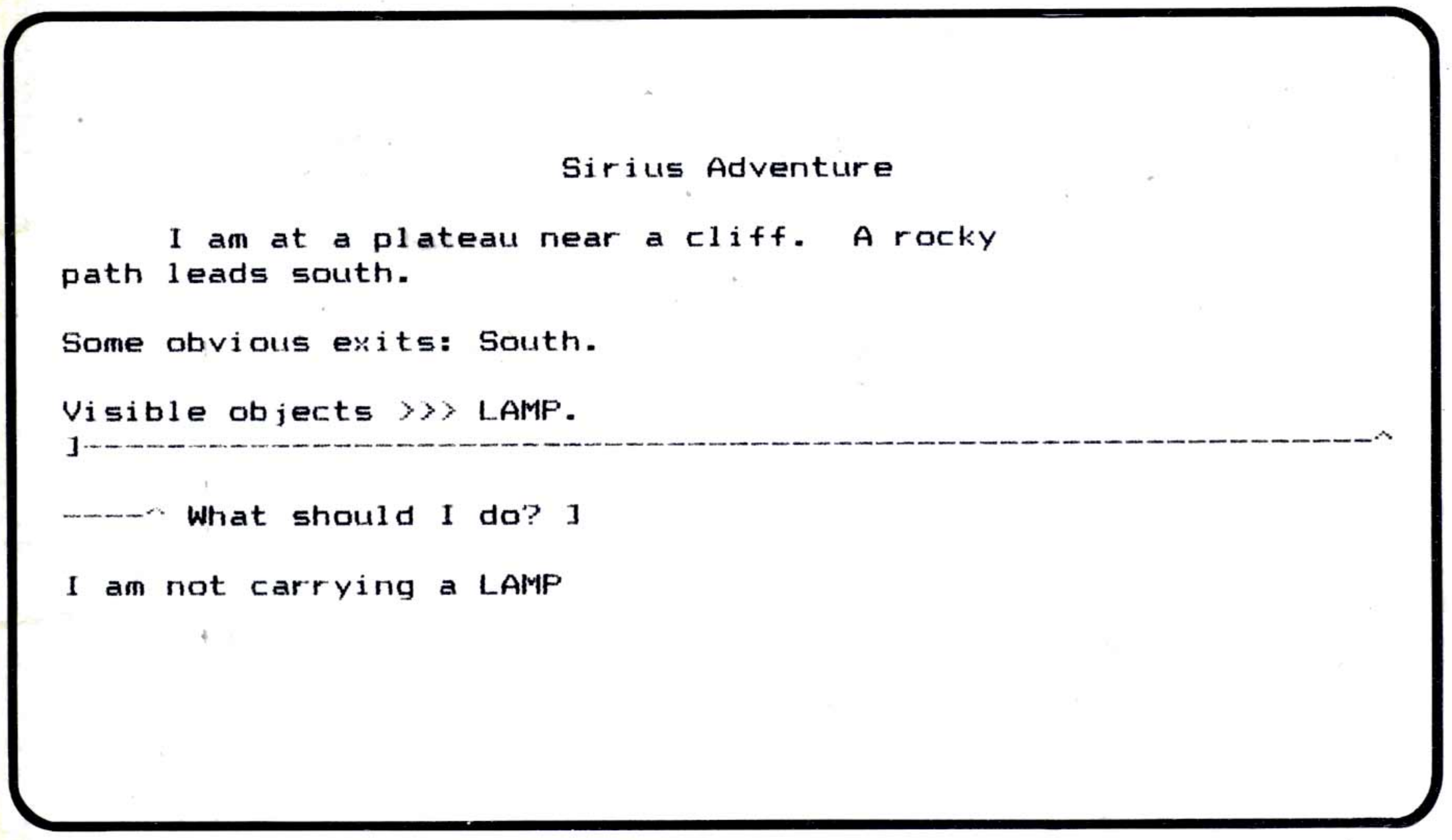
MICRO-80

P.O. BOX 213, GOODWOOD, S.A. 5034. AUSTRALIA. TELEPHONE (08) 211 7244. PRICE: AUS. $3.50. N.Z. $5.00. U.K. £1.50
Registered by Australia Post – Publication No. SBQ2207

Vol. 4, Issue 7, 1984

INSIDE: PROGRAMS FOR THE VZ 200



```
                    Sirius Adventure

     I am at a plateau near a cliff.  A rocky
path leads south.

Some obvious exits: South.

Visible objects >>> LAMP.
]----------------------------------------------------------^

-----^ What should I do? ]

I am not carrying a LAMP
```

Also in this issue:

Improvements to OS 80 Operating System
Three New VZ 200 Commands
High Score Graphics Routine for CoCo

SOFTWARE:
Dogfight (CoCo)
Disk Directory Recorder (Model 3)
Sharemarket (Level II)
Words and Meanings (Level II)
Array Utility (Level II)
Junior Maths (VZ 200)
Battleships (VZ 200)

• TRS-80 • SYSTEM 80 • VIDEO GENIE
• PMC-80 • VZ 200
• TRS-80 COLOUR COMPUTER

ANNOUNCING THE '80 XT EXPANSION FOR SYSTEM 80 AND TRS-80 COMPUTERS FROM $1,199

DISK CONTROLLER, 32K RAM AND TWO DISK DRIVES ALL IN THE ONE ATTRACTIVE, COMPACT CABINET

The TRS-80/System 80 computer when equipped with additional memory and disk drives is still one of the most versatile and powerful home computer systems available. It makes a powerful word processor or data base manager which can be used in serious applications. If you would like to increase your computing power and experience economically with proven equipment and software, you should seriously consider upgrading your L2/16K machine by the addition of the appropriate '80 XT expansion.

XT stands for EXTRA and MICRO-80's '80 XT has plenty of extras. The one attractive, vinyl covered metal cabinet houses:

- ☐ Two slimline disk drives of 100K, 200K or 400K capacity each.
- ☐ A heavy duty switching power supply to give cool, reliable operation free from power glitches and random "reboots".
- ☐ DOSPLUS 3.5 disk operating system.
- ☐ MICRO-80's proven expansion interface board giving:
  - up to 32K static ram: to ensure high noise immunity and reliability
  - single density disk controller: for complete compatability with all disk operating systems
  - centronics printer port: the system 80 model has a double-decoded port to respond to both port FD and memory address 37E8H thus overcoming one of the major incompatabilities with the TRS-80.
  - RS232 communications port: for communicating by modem or direct link to other computers
  - real time clock interrupt: provides software clock facility used by most DOS's

Economical double density: an economical, high quality double density upgrade will be released shortly to enable you to increase the capacity of your disk drives by 80%.

THE INTEGRATED DESIGN OF THE '80 XT SAVES YOU MONEY TOO:

'80 XT WITH OK RAM AND TWO SINGLE—SIDE 40 TRACK DISK DRIVES (100K byte each)	**$1,199**
'80 XT WITH OK RAM AND TWO DOUBLE-SIDE 40 TRACK DISK DRIVES (200K byte each)	**$1,299**
'80 XT WITH OK RAM AND TWO DOUBLE-SIDE 80 TRACK DISK DRIVES (400K byte each)	**$1,499**

ADDITIONAL 16K RAM **$99** ADDITIONAL 32K RAM **$198**

All configurations available ex stock NOW
Be sure to specify whether you have a TRS-80 MODEL 1 or a SYSTEM 80.
Add $12.00 delivery anywhere in Australia.

CONTENTS

ABOUT MICRO-80

EDITOR: IAN VAGG

MICRO-80 is an international magazine devoted to the Tandy TRS-80 Model 1, Model III and Colour microcomputers, the Dick Smith System 80/Video Genie and the Hitachi Peach. It is available at the following prices:

	12 Months	Single Copy
MAGAZINE ONLY	$ 36.00	$ 3.50
CASSETTE SUBSCRIPTION	$ 96.00	$ 6.00
DISK SUBSCRIPTION	$125.00	$10.00 (disk)

MICRO-80 is available in the United Kingdom from:
U.K. SUBSCRIPTION DEPT. 24 Woodhill Park, Pembury, Turnbridge Wells, KENT TN2 4NW

MAGAZINE ONLY	£16.00	£1.50
CASSETTE SUBSCRIPTION	£43.00	£N/A
DISK SUBSCRIPTION	£75.00	£N/A

MICRO-80 is available in New Zealand from:
MICRO PROCESSOR SERVICES, 940A Columbo Street, CHRISTCHURCH 1 NZ. Ph. 62894

MAGAZINE ONLY	NZ$ 59.00	NZ$ 5.60
CASSETTE SUBSCRIPTION	NZ$130.00	NZ$ 7.50
DISK SUBSCRIPTION	NZ$175.00	NZ$15.00

MICRO-80 is despatched from Australia by airmail to other countries at the following rates:

(12 MONTH SUB)	Magazine	Cass Sub	Disk Sub
PAPUA NEW GUINEA	Aus$53.50	Aus$115.50	Aus$148.50
HONG KONG/SINGAPORE	Aus$58.00	Aus$122.00	Aus$157.50
INDIA/JAPAN	Aus$64.00	Aus$129.00	Aus$165.00
USA/MIDDLE EAST/CANADA	Aus$73.00	Aus$140.00	Aus$177.00

Special bulk purchase rates are also available to computer shops etc. Please use the form in this issue to order your copy or subscription.

The purpose of MICRO-80 is to publish software and other information to help you get the most from your TRS-80, System 80/Video Genie or Peach and its peripherals. MICRO-80 is in no way connected with any of the Tandy, Dick Smith or Hitachi organisations.

WE WILL PAY YOU TO PUBLISH YOUR PROGRAMS: Most of the information we publish is provided by our readers, to whom we pay royalties. An application form containing full details of how you can use your microcomputer to earn some extra income is included in every issue.

CONTENT: Each month we publish at least one applications program in BASIC for each of the microcomputers we support. We also publish Utility programs in BASIC and Machine Language. We publish articles on hardware modifications, constructional articles for useful peripherals, articles on programming techniques both in Assembly Language and BASIC, new product reviews for both hardware and software and we printer letters to the Editor.

LIABILITY: The programs and other articles in MICRO-80 are published in good faith and we do our utmost to ensure that they function as described. However, no liability can be accepted for the failure of any program or other article to function satisfactorily or for any consequential damages arising from their use for any purpose whatsoever.

MICRO-80 is Registered by Australia Post — Publication No. SBQ2207

AUSTRALIAN OFFICE AND EDITOR: MICRO-80, P.O. Box 213, Goodwood, S.A. 5034. Tel. (08) 211 7244

U.K. SUBSCRIPTION DEPARTMENT: 24 Woodhill Park, Pembury, Turnbridge Wells, Kent TN2 4NW

TYPESETTING & MAKE-UP: Formgraphic, 117 Wright Street, Adelaide, S.A. 5000. Tel. (08) 211 7866

PRINTED BY: Specialty Printers, 42 Wodonga Street, Beverley, S.A. 5009

PUBLISHED IN AUSTRALIA BY: MICRO-80, 433 Morphett Street, Adelaide, S.A. 5000

EDITORIAL

This issue we welcome the VZ200 users to our columns. The VZ200 is an interesting machine. It fits in well with our '80's because of its Z80A processor whilst having some of the attributes of the CoCo. We have also been impressed by the quality and low price of much of the arcade game software available for the VZed. For the Editorial Staff at Micro-80, using a VZed is like turning the clock back 5 years. As yet, there are no disk drives available and the amount of information concerning the inner workings of the computer is sparse to say the least. No Editor/Assembler has yet appeared nor is there a command to allow you to load in machine language programs directly. Clearly, there is a great deal of fascinating exploration to be carried out which we shall enjoy being involved in as much as our readers.

If you are a VZed owner reading MICRO-80 for the first time, welcome. Please write to us seeking or sharing information. Also, send in the programs you have developed for which we will pay you a fee on publication. Articles, reviews of relevant products etc. are also welcome. Many of you will be relative newcomers to computing. We shall cater to your needs with basic articles on programming and explanations of how your computer works. We hope you enjoy being one of us.

From America comes the news that OMIKRON is in serious financial difficulty and may need to close down. OMIKRON is a somewhat unusual organisation which is not well known in Australia. Nevertheless, it has been involved with TRS-80's since the earliest days of the Model I. OMIKRON was the first company to make the CP/M operating system and 8 inch disk drives available on the TRS-80. The company has backed CP/M heavily and has sold application programs under CP/M at very low prices. It is this attempt to bring low cost software to the TRS-80 user which has brought on OMIKRON's troubles. The company has committed itself to pay over a quarter of a million dollars for software licences. Unfortunately, it has been unable to sell the necessary volume of software to meet these committments. The lack of funds has, in turn, prevented it from developing new products. It is a classic case of selling things too cheaply. Now, OMIKRON has launched what is effectively an appeal to all the owners of OMIKRON mapper boards (i.e. CP/M adaptors for TRS-80's) to buy an item of software at what is virtually a give away price in an attempt to generate this badly needed cash flow. For $39 US you can buy an Accounting System module (e.g. Accounts Receivable or General Ledger etc.) or the Tarbell Data Base or Electric Webster Spelling Checker and so on. We wish OMIKRON luck in this rescue attempt as the company has certainly made a valuable contribution to TRS-80 users.

This month we have a very full issue with lots of programs and readers letters to answer. It is clear from the mail we receive that the INPUT/OUTPUT column is extremely popular. If you know the solution to a problem voiced in these pages, don't hesitate to write in. One of MICRO-80's objectives is to help readers help each other.

DEPARTMENTS

KALEIDOSCOPE

When you are writing games using the high resolution screen, it is often necessary to present the score to the player as the game progresses. In high resolution mode, this must be done graphically. To help you along, Charlie Bartlett wrote a subroutine which you can include in any of your programs (why not use the Merge routine published last issue to add it to existing games).

```
**** HIGH RES SCREEN SCORE ****

        COLOUR COMPUTER


10 ' HIGH RESOLUTION SCREEN
     SCORE SUBROUTINE
     (C) 1983  C.BARTLETT

20 CLEAR500:PMODE3,1:PCLS(2):SCR
EEN1,0:GOSUB32000
40 SC=9990
50 SC=SC+1:GOSUB32130
60 FOR T=1TO500:NEXT T:GOTO50
70 '
80 '
90 '
100 '
32000 ' SUBROUTINE TO LOAD IN
32010 ' NUMBER STRINGS

32020 N$(1)="BR4BD1E1D6L1R2BU6"
32030 N$(2)="BR4BD1E1R2F1D1G3L1D
1R4BU6"
32040 N$(3)="BR4BD1E1R2F1D1G1F1D
1G1L2H1BR5BU5"
32050 N$(4)="BR6D6U3R2L5E3BR2"
32060 N$(5)="BR5R4L4D2R3F1D2G1L2
H1BR4BU5"
32070 N$(6)="BR6L1G1D4F1R2E1U1H1
L3BR4BU3"
32080 N$(7)="BR4R4D1G2D1G2BR4BU6
"
32090 N$(8)="BR5R2F1D1G1L2G1D1F1
R2E1U1H1L2H1U1E1BR3"
32100 N$(9)="BR4BD1E1R2F1D4G1L2H
1BU4D1F1R3BU3"
32110 N$(0)="BR5R2F1D4G1L2H1U4E1
BR3"
32120 RETURN
32130 '
        SUBROUTINE TO DRAW
32140 ' GRAPHIC NUMBERS

32150 COLOR2:DRAW"S8BM100,100;XK
B$;":IF SC <10THENSS$="000"+RIGH
T$(STR$(SC),1):GOTO32190
32160 IF SC <100THENSS$="00"+RIG
HT$(STR$(SC),2):GOTO32190
32170 IF SC <1000THENSS$="0"+RIG
HT$(STR$(SC),3):GOTO32190
32180 SS$=RIGHT$(STR$(SC),4)
32190 B1=VAL(LEFT$(SS$,1))
32200 B2=VAL(MID$(SS$,2,1))
32210 B3=VAL(MID$(SS$,3,1))
32220 B4=VAL(RIGHT$(SS$,1))
32230 KB$=N$(B1)+N$(B2)+N$(B3)+N
$(B4)
32240 COLOR3:DRAW"S8BM100,100;XK
B$;"
32250 RETURN
```

This subroutine can be used with any program that requires a score display on a high resolution screen. The user simply initializes the number strings by — GOSUB32000 — at the start of his program, the variable "SC" should be used in the users program to keep count of the score. The main subroutine at line 32130 will take the value found in the variable "SC" and turn it into the proper high resolution equivalent.

Line 32150 is used to blank out the old score, the statement COLOR 2 should be set to the background colour if a different colour set is chosen. In this line and line 32240 the scale function of the DRAW command has been set to 8. The scale can be set as low as 4 in PMODEs 3 and 4 and the display will still be clear. If you change the scale or position statements in one line you must change them in the other line as well.

The lines following this, up to line 32170 are used to convert the variable "SC" to a string and pad it on the left with the required number of zeroes.

Lines 32190 to 32220 assign each position in the string to a variable, i.e.: leftmost character is assigned to variable "B1" the following character to variable "B2" etc.

Lines 32230 builds the display string and line 32240 displays it. Its as simple as that, so as long as you use "SC" to keep score, (or change the subroutine), all you have to do to have a high resolution screen score is put the statement — GOSUB 32130 in your program each time that the score is updated.

FORM ONE

Users of '80 computers equipped with disk drives are particularly fortunate in the range of DOS's available to them. The more powerful systems such as DOSPLUS, LDOS and NEWDOS (here arranged in alphabetical order to avoid charges of favouratism!!) have operating systems superior in many respects to those available on the new crop of 16 bit Micro's. One of the less well known and simpler operating systems is OS-80. This system was developed by PERCOM in the U.S.A. and sold in Australasia by Dick Smith Electronics. We have said very little about OS-80 in Micro-80 for one very good reason, we don't know very much about it ourselves!! Nevertheless, it is certain that many of our readers have used it or are still using it on their System-80 computers. Despite the fact that it has now been discontinued by PERCOM or perhaps because of that we feel that the following contribution from Barry Briggs of 14 Allenberry Ave., Napier HB, New Zealand will be of value to them.

Dear Sir,

I'm writing to you in reply to a query by D. Sutton, (Micro 80, Vol. 4, Issue 6, Page 7) concerning saving System Tapes with OS-80. About a year ago a friend of mine had a similar problem, only his complaint was that Dick Smith's patch was a Basic program and saved M/L files as strings. It worked well, BUT it meant that if he had a Basic program resident and he wanted to load in a Utility program, he had to save the program he was working on, load the Utility and then reload the original program.

He asked me if I could do any thing about the problem, and, not knowing what was in store, I agreed to give it a go. I borrowed a Disk Drive and Expansion Interface (I now have my own drive) and got into the Dos. A year and a few grey hairs later and after much testing, rewriting, debugging and having a lot of fun in the process, Enhanced OS-80 came to life.

At the present time I have sold several copies (by word of mouth) and am considering advertising to see what eventuates. (I don't know how many copies of OS-80 there are in NZ, but it's worth a go). If you feel that this patch could be the answer to Mr Suttons problem perhaps you could pass this letter and the attached 'Blurb' on to him.

I realise that by this time he has probably bought a more sophisticated DOS, however in the process of learning about OS-80, I came to appreciate its simplicity, coupled with its speed when it comes to DATA handling. Also for those with a single drive, more can be put onto a Disk, a 250 byte program on OS-80 uses one sector, on all other Dos's it would take 1 Gran or 5 sectors.

For these reasons, for some applications OS-80 is an excellent DOS, made more so by it not being Disk dependent, another saving in space. (Data disks on a single Drive!!)

ENHANCED OS-80

Enhanced OS-80 is a modified version of OS-80, adding commands that increase the capability of the Dos. Listed below are the extra commands available. If (#) follows a description, then this command may be part of a BASIC program.

Note: Machine language files saved with this Dos are -not- compatable with the standard Dos.

- Load (#) and save machine language files as machine language files, that means no BASIC loader to overwrite a BASIC program already in memory.
- Load system tapes and display name and parameters.
- Add an offset to tapes to prevent conflict with Dos. If the tape is offset then a block move appendage may be added to move the file back to its correct location when reloaded from disk.
- Set a new mem size as a direct command. (#).
- Lower case driver that is selective, if your machine doesn't have l/c it won't attempt to print l/c.
- For those difficult programs that seem to be a mixture of caps and l/c, the lowercase driver may be disabled. (#).
- Turn cursor flash off and on from keyboard. (#).
- Renumber or check program for undefined line numbers.
- Remove unwanted spaces (and rems) from a BASIC program.
- Restore a NEW'ed program.
- Calculate and display length and sectors required for a BASIC program in memory. Also called when writing a m/l file to disk.
- Toggle between (SHIFT) caps or (SHIFT) lowercase.
- Route to printer and screen (#).
- 'No hang' printer driver patch (uses Rom printer driver).
- Dos sectors (0-29) have a software write protect to prevent overwriting them by mistake. The sectors protected may be altered if desired. (Holding (SHIFT) allows writing to these sectors without altering the Dos).
- In keeping with the original concept of OS-80, enhanced OS-80 is non disk dependant. (Once booted the disk may be removed).
- Entire enhanced Dos is copied with CMD'M' or CMD'I'.

On the debit side these features take up more room on your disk. The Dos now uses sectors 0 to 29 (10 more than normal) and the start of BASIC is now 6400H not 5A00H as before. However the advantages far outnumber the disadvantages as can be seen.

Supplied with full instructions (as a BASIC program) plus a configuring program to enable changes to key repeat, cursor flash rate, cursor character (plus other changes) to be made simply and easily.

Conditions of Sale . . .

To avoid copyright problems, the purchaser must supply a formatted disk containing OS-80. This will be 'zapped' and returned along with full instructions (covering the extras only).

The price of these enhancements is $20.00 (N.Z.). Post Free.

OS-80 and Microdos and trademarks of Percom Data Co.

Command Summary — Refer to BASIC program for expanded details.

Save DSSSS	Save a basic file.
Save @ DSSSS	Save a machine language file with optional tape load, offset and block move appendage. Hold (SHIFT) to access sectors 0-29.
Load DSSS (,R)	Load or run a basic file.
Load @ DSSSS (,R)	Load or run a machine language file holding (SHIFT) will display parameters.
CMD"B",XXXXX	Set mem size and clear 50. (XXXXX may be a numeric expression).
CMD"S"	Calculate and display program length plus sectors required.
CMD"L",Y	Enable lower case driver.
CMD"L",N	Disable lower case driver.
CMD"L",P	Route to printer. Enter 'CMD"L", (Y or N) or (BREAK) to cancel command.
CMD"R",N,O,I	Renumber (N = Newline, O = Oldline, I = INC).
CMD"R",C	Check for undefined lines.
CMD"C",S	Remove unwanted spaces.
CMD"C",R	Remove unwanted spaces and rems.
CMD"Z"	Restore 'New'Ed program, results may be unpredictable if keyboard entry made between 'New' and CMD"Z".
Print CHR$(2) or (SHIFT)(DOWN ARROW) and 'B'	Turn cursor and key repeat on.
Print CHR$(3) or (SHIFT)(DOWN ARROW) and 'C'	Turn cursor and key repeat off.
(SHIFT) 'O'	Toggle between (SHIFT) lowercase and (SHIFT) caps.

Numerical input when writing a m/l file to disk, may be decimal or hex. (preceed hex with '&H'). Setting mem size may be decimal, hex or an expression e.g. 'INPUT A: if A)32767 then A = A-65536:CMD"B", A'.

V-ZED — THREE NEW FUNCTIONS

This is a regular feature to assist VZ 200 users to come to understand more about their computers and to learn a few tricks which are not necessarily covered by the manuals. We welcome contributions from Readers who have discovered new features of the machine or interesting techniques which they would like to share with their fellow VZ-200 users.

The BASIC Interpreter in the VZ 200 was written by MICROSOFT, the company which developed the first BASIC Interpreter for a microcomputer way back in the mid 70's and which probably supplies over 80% of all BASIC Interpreters in use today. Not surprisingly, when a new computer such as the VZ comes along, MICROSOFT takes its standard BASIC Interpreter and modifies it to suit the new hardware and the particular features which the manufacturer would like included. From the user's point of view there are both advantages and disadvantages to this approach. The main disadvantage is that the resulting code can become very untidy with patches on patches right throughout the ROM. The outcome often being inefficient use of space and slower execution. On the positive side however, there are likely to be routines still left in from other interpreters which are not intended to be available in the VZ but, with a little fiddling can be used. To the average computer user, the thrill of making your computer do something which the manufacturer never intended, is worth any of the disadvantages. The purpose of this article is to start you off with three hidden functions. Once you start experimenting in this area you will no doubt find others. Please write in and let us know about them so that we may all share in them.

The MICROSOFT BASIC interpreter as implemented in the Tandy TRS-80 Model 1 occupied 12 Kbytes of ROM. Although we do not know for

sure, it is likely that this implementation started a new family of BASIC interpreters of which the VZ's is a derivative. Certainly there seems to be no surplus code in the Tandy Interpreter although the Model 3 version shows evidence of having been extensively patched and hacked around. The interpreter in the VZ has a number of additional features over and above those available in the Tandy. In particular, the support for higher screen resolution, colour and full screen editing obviously requires extra code. Even though this interpreter now occupies 16K of ROM it became necessary to leave out some of the features which had been in the TRS-80 version. In particular, the AUTO TRACE function and the free memory indicator have gone whilst there is no facility to turn off the sound, should you wish to do so. However, the essential routines to do all these things remain locked away in the ROM and can be accessed with a bit of judicious POKEing.

AUTO LINE NUMBERING

The interpreter contains an AUTO line numbering routine which when activated, automatically prints the next line number on the screen to speed up the entry of BASIC programs. It is possible to specify the starting line number and the increment between line numbers. For example, you may wish to start entering lines commencing with line 100 with an increment of 10 so that the second line would be 110 the third 120 etc. The AUTO routine operates every time you press the RETURN key from the COMMAND mode. It looks at address 30945. If that address contains a zero then AUTO numbering is off and the computer behaves normally. However, if that value is 1, the AUTO routine looks at addresses 30946 and 30947 to find the value of the starting line number then at addresses 30948 and 30949 for the increment between line numbers. The next line number is then automatically displayed on the screen. The only part of the AUTO routines missing is the ability to recognise the AUTO command itself. However, if you POKE the appropriate values into the memory addresses above, you will be able to use this facility.

To set the starting line number, POKE the decimal equivalent of its Least Significant Byte (LSB) into address 30946 and the decimal equivalent of its Most Significant Byte (MSB) into 30947. Similarly, to set the line increment, POKE its LSB into 30948 and its MSB into 30949. It is likely that this is double Dutch to relatively new users of the VZ so we have illustrated the techniques with the program below. If you wish to know more about the subject of POKEing etc. you will find a good article in Volume 4, issue 4/5.

We suggest you enter this routine, make sure it works satisfactorily then CSAVE it under the name AUTO or similar. You can then load it in whenever you are doing program development. We have used high line numbers to keep it out of the way of your own programs. To start it operating, type RUN 60,000. Incidentally, you terminate AUTO line numbering by pressing the BREAK key.

PROGRAM LISTING 1

```
60000 REM SET STARTING LINE NO                FOR THE AUTO ROUTINE
60010 INPUT"STARTING LINE NUMBER";SL
60020 POKE 30946,(SL-256*INT(SL/256))
60030 POKE 30947,INT(SL/256)
60050 REM SET THE INCREMENT                   BETWEEN LINE NUMBERS
60060 INPUT"INCREMENT BETWEEN LINE NOS";IN
60070 POKE 30948,(IN-256*INT(IN/256))
60080 POKE 30949,INT(IN/256)
60100 REM SWITCH ON THE AUTO                  LINE NUMBERING ROUTINE
60110 POKE30945,1
```

TURNING OFF THE BEEPING KEYBOARD

Now that you have AUTO line numbering, you will probably want to sit up all night entering programs. Only trouble is, the beeping of the keys is likely to keep the rest of the family awake.

No problem:
POKE 30779, 0 disables the key beep whilst
POKE 30779, 1 turns it on again.

You may enter this straight from the keyboard or include it as a line in your program.

Incidentally, this memory address appears to carry out some other functions, depending on the bit that is set. We did a little experimenting and found that bit 0 turns on and off the beep as expected i.e. an even value POKEd into address 30779 turns off the beep whilst an odd number turns it on i.e. 0, 2, 4, 6, 8 etc. turn it off, 1, 3, 5, 7, 9 etc. turn it on. Bits 1 and 2 have no special effect but bit 3 clears the screen and positions the cursor at the bottom left hand corner. This bit also causes an audible click from somewhere inside the computer probably from the piezo electric speaker. Bit 4 changes the background colour from green to orange. As far as we could tell bits 5, 6 and 7 had no effect.

FREE SPACE

Probably the most useful POKE for a programmer would be a way of finding out how much string space is available or how much memory you have left to cram in those last few lines before being told by the machine that you are Out of Memory.
Try the following.

```
POKE 30862,212:
POKE 30863,39:
PRINT USR(X) 'FREE MEMORY
  OR
PRINT USR(X$) 'FREE STRING
SPACE
```

SOFTWARE

ARRAY UTILITY (L2/16K) version 2 (Oct. 81)

by R.E. Taplin

PROGRAM TO RECORD, LOAD, ERASE OR RENAME AN ARRAY

This program will be of interest to TRS-80 users who have only a cassette system for storing numerical or string data. It enables the transfer of arrays between RAM and tape in a fashion independent of their original locations in memory and at a rate that is limited primarily by the baud rate of the cassette system. The program also provides for the erasure or renaming of numeric or string arrays, two procedures which can be used to optimize the use of available memory.

***TO RECORD AN ARRAY: the statement

SAVE array name

may be placed at any appropriate point in a Basic program or used in Command mode following the running of a program to recover a desired array.

The array name may have any form acceptable to Level II Basic. Note that only the array name itself is used. No brackets or array element indices are permitted. In usual **Basic** fashion, blanks are ignored.

Execution of the command begins with a search for the array in memory. When it is located the operator is warned to prepare the recorder by a READY CASSETTE message. At this point, pressing the BREAK key will abort the command and control will revert to the next instruction in the parent Basic program. Pressing any other key results in the array being dumped to tape followed by a checksum.

String arrays are recorded differently from numeric arrays because of the distribution of string array information between the **Basic** array table and string space. To assist the programmer to allow sufficient string space when reloading a recorded string array, the total number of characters recorded is displayed on the screen at the end of each string SAVE.

The program is initialized to use cassette #1. If the operator wishes to use cassette #2, a non-zero value must be POKed into 16446.

***TO LOAD A RECORDED ARRAY: the instruction

LOAD array name

is used. Any array name APPROPRIATE TO THE TYPE of the recorded array may be specified in the LOAD instruction. The new name is substituted for the old array name once the load is completed. (If an array name of inappropriate type is specified, **Basic** ignores the loaded array and creates a new array when the name is encountered in a subsequent array operation. There may be the rare occasion when one could exploit this feature, but

normally one would want to avoid it.)

As with the SAVE instruction, the operator can abort the procedure when the READY CASSETTE warning is given. Control passes to the next **Basic** instruction.

No prior dimensioning is necessary for a loaded array as the LOAD procedure itself accomplishes this. (The effect of prior dimensioning is to create an array of the same name and type as the loaded array in the array table. Because it is earlier in the table, **Basic** accesses it rather than the loaded array whenever the array is called in the remainder of the program.) However, some programs require the dimensioning of an array that may be LOADed at a later stage. For example, a data management program may provide for data entry, saving of data to tape, and loading of data from tape. The data entry rountine would require that an array be dimensioned, whereas the LOAD routine would not. This conflicting requirement can be readily overcome by either confining the array dimensioning to the data entry routine or, if that is undesirable, by the judicious use of array erasure using KILL.
e.g. A program may have the form:

```
- - - - -
DIM D$(500) 'Dimension data array
'Data entry routine
- - - - -
SAVE D$ 'Record data
- - - - -
'Load data routine
KILL D$
LOAD D$
- - - - -
```

The instruction KILL coming just prior to LOAD erases the earlier version of D$, leaving the array table free for the new version. (But see the section on KIll for discussion of the strong space cost in using this technique.)

The operator can monitor the LOAD via the usual flashing asterisk in the top right-hand corner of the screen. For numeric arrays the asterisk flashes once for every 256 bytes. For string arrays it flashes after the entry of each array element. The operator will find, therefore, that there may be no flashing for small numerical arrays.

Invalid loads are detected by means of a conventional checksum at the end of the load. If there is a checksum error the message SAVE/LOAD ERROR is displayed on the screen, and the bytes added to the Basic array table are discarded. In the case of string arrays, the string space pointer is reset to its initial value.

A message giving the assigned name of the loaded array and its dimensions and type is available to the operator at the end of a load should he or she desire it. The program is initialized to bypass the message, but it may be obtained by POKEing a non-zero value into 16447. Because control returns to the next **Basic** statement after a LOAD it may be necessary to temporarily halt the program with an INPUT in order to inspect it. The message has the format:
ARRAY: NM TYP DIM: N1 N2 N3 . . .
where NM are the first 2 letters of the array name, TYP is one of STR, INT, SNG, or DBL, depending on the type of the array, and Ni, N2, etc. are the depths of the successive dimensions. The type identifiers: $ % ! # are omitted from the name. For example, the two dimension string array A$ dimensioned by DIMA$ (50,2) would give the message:

ARRAY: A STR DIM: 50 2

When LOADing string arrays, the operator is responsible for clearing sufficient string space. (See the comment in the SAVE section.) If insufficient string space is available for inserting strings, the LOAD will be oborted and the message

OUT OF STRING SPACE

is displayed on the screen. Array and string space pointers are returned to their initial values.

***TO ERASE AN ARRAY: use

KILL array name

This command is particularly useful when working with a series of large arrays and limited memory. It has the effect of discarding the unwanted array and moving the remainder of the array table into the locations the array held in memory. Unfortunately, in the case of string arrays, while KILL removes the array details from the array table, it does not remove the array's strings from the string space. This limitation necessarily arises from the irregular way in which array element strings may be distributed throughout string space. If the programmer is willing to sacrifice all strings created up until the erasure, he can recover string space by resetting the string space pointer at 40D6H (16598) to MEMORY SIZE.
e.g. use:
POKE 16598,PEEK(16561):
POKE16599,PEEK(16562)

***TO CHANGE THE NAME OF AN ARRAY: one may use the command
NAME old array name, new array name

This procedure is another time and memory saver as it makes possible the use of a single **Basic** subroutine for processing a succession of arrays, without the clumsiness of having to assign each array, element by element, to a general purpose subroutine array. This facility brings the **Basic** programmer a fraction closer to the convenience of the parameterized procedures of languages such as FORTRAN or PASCAL. The procedure also swaps the type codes for the old and new names in the Mode Table at 4101H, allowing the programmer to ignore array type differences. This means that the one subroutine may be used with single or double precision numerical arrays, or even with string arrays. For example: Arrays with names starting with N may be defined in a program as integer using the DEFINT verb. Assuming that the letter G retains its initialized type of single precision, the procedure NAMENA,GP would substitute the name GP for the name NA in the array 'NA', and ALSO swap the integer code 2 with the single precision code 4 for the letters N and G in the Mode Table. Thus, at the end of the procedure the letter N would have the code 4 and G the code 2. This provision carries with it the danger that other variables may be affected by the change, but its advantages should outweigh its liabilities. In any case, the programmer can minimize the danger of type confusions by reversing a name change immediately after the need for it has passed.

In addition to the error messages explained above, the program also provides the following:
EMPTY TABLE When no arrays are currently dimensioned.
NOT FOUND When no array of the name specified in a SAVE, KILL or NAME statement is currently dimensioned.

ADDRESSES for the program ARRAY:

	START	END	ENTRY	
16K	7BAB	7FFE	7BAB	(31659)
32K	BBAB	BFFE	BBAB	(48043)
48K	FBAB	FFFE	FBAB	(64427)

WORDS AND MEANINGS (L2/16K)

by Murray J. Dixon

This program is designed to assist students with difficulties in basic English, but it could find other uses in areas where a knowledge of definitions is required.

From a list of data, the program reads both words and definitions into an array. It then prints the words in a random order at the top of the screen. One of the definitions is then printed and the user is required to type the correct corresponding word from the list. The computer will continue to ask the question until it receives a correct response.

The data list can be readily extended or altered to suit the particular level or application, however the total number of data pairs must be placed in the variable DD in line 20.

The words and definitions are read into the two-dimensional A $ array, checking for non repetition (lines 140-250), then the words are printed at the top of the screen in a random order (lines 270-350). The definition to be matched is then chosen randomly from the array and printed on the screen (lines 390-490). The user entered response is compared with the correct response in line 500. The variable R is a counter for the number of incorrect responses on the first attempt.

SHAREMARKET (L2/16K)

by R.J. Burling

The programme is based on the popular game of Stockmarket. Similar boundaries for upper and lower share prices are fixed within the program. The share prices are independently and randomly moved (within given parameters) varying from all up 10 points right through to all down 10 points. Penalties are also there.

The program proper commences by determining the number of players (1-4) and obtaining their names. Then depending on the number, moves

through a preselected number of turns before asking whether to finish or continue. If continue is the choice then all the random variables are randomised whilst 'The market is being studied'. Then the game continues. Some entries use the Inkey$ function whilst others require you to press the ENTER/NEW LINE key.

When penalties or bonuses are incurred, the advancement to the next turn is automatic (after a time lag).

If players overdraw their accounts they have the option of selling shares of their choice or liquidating.

For each player, the shares, their current value, the number held, the bank balance, the assets and the total number of shares held are displayed along with the particular action for that turn.

At the end of the game each players assets will be displayed.

DOGFIGHT (16K Coco)

by Stephen Gibbons

Dogfight is a game for two players. It will run on any standard 16K colour computer.

The object of the game is to shoot down your opponent before he shoots you.

The first pilot to shoot down his opponent ten times wins.

You duel over mountains which are randomly shaped, so they are different every time. Don't fly too close to the mountains. If you hit one you will explode.

Flying out of the left side of the screen will result in appearing at the right side of the screen and vice versa, but you cannot fly out of the top of the screen. If you try to do this you will automatically change direction.

If you fly into your opponent or your opponent flys into you, both you and your opponent will explode.

Steering can be controlled by the keyboard or joysticks if you have them.

There are eight directions that each plane can go. They are up, down, left and right as well as four diagonal directions.

The controls are as follows:

Left Player:

(Q) To change direction one position anticlockwise.
(W) To change direction one position clockwise.
(Up Arrow) To fire machine guns.

Right Player:

(Left Arrow) To change direction one position anticlockwise.
(Right Arrow) To change direction one position clockwise.
(@) To fire machine guns.

If you have joysticks, push the joysticks lever left to change direction one position anticlockwise or right to change direction one position clockwise or hit the fire button to fire machine guns.

To shoot your opponent down, get on his tail and hit the fire button. A white bullet will advance five spaces ahead of your plane in the direction that you are going. If the bullet hits your opponents plane or even misses by one graphics block, you will see his plane dive into the mountains leaving a trail of smoke then explode in a shower of sparks.

May the best Baron win!

SIRIUS ADVENTURE (L2/16K)

by M. Laden Bauk

The adventure takes 8.5K of memory (even less if packed). It is a very basic adventure module which I wrote in a structured way in order for it to be easily altered and expanded. New verbs, nouns and locations can be added with a minimum of alterations to the existing program. At present, the adventure understands verbs when they are applied only to objects, (i.e. 'LOOK LAMP', 'EAT LAMP' etc.) with the exception of the 'GO' command. A breakdown of the program follows.

PROGRAM STRUCTURE

LINE NO.	DESCRIPTION
200-240	Initialisation of variables: *VB → No. of verbs *ND → No. of nouns/directions *L → No. of locations *OB → No. of objects
310-410	Screen update routine.
420-460	Manipulate user input, LE$ = LEFT HAND WORD (1st WORD) RI$ = RIGHT HAND WORD (2nd WORD)
470-500	Test for 1st word (verb).
510-540	Test for 2nd word (noun/direction).
560	Program flow diverted according to verb used.
570	If verb was 'EAT', 'GET' type or 'DROP' type then update screen.
580	If verb was 'VOCABULARY' then update screen.
590	Make sure 2nd word isn't an object.
600	Divert program flow according to the direction adventurer has specified.
610-1040	Move in direction, if possible.
1060-1090	Eat <OBJECT> routine.
1110-1160	Get <OBJECT> routine.
1190-1210	Drop <OBJECT> routine.
1230-1290	Look <OBJECT> routine.
1320-1380	Wave <OBJECT> routine.
1410-1440	Quit <OBJECT> routine.
1460-1520	Score routine.
1530-1580	Inventory routine.
1600-1740	Save/Load routine (Disk only).
1760-2040	Initialisation routine (variables).
2060-2250	Instruction routine.
2260-2290	Obstruction routines.
2300-2420	Keyboard input routine (eat your heart out Ken).

DIRECTIONS FOR EXTENDING THE ADVENTURE

In the program: 'I' holds the positional value of a verb in the verb list (line 1790) and 'J' holds the positional value of a noun in the noun list (line 1800).

ADDING AN OBJECT:

In line 240 increment OB (objects) by one (OB = 7).
In line 1800 append new object's description to list.
In line 1810 append new object's location to list.

ADDING A NEW VERB

If the object added was a box and an 'OPEN' command is required, then:
In line 240 increment VB (verbs) by one.
In line 560 append (i.e. after No. 2400) a new line to handle 'OPEN' routine. For example, line 2430.
In line 1790 append the word 'OPEN' to list.
In line 2430 write the 'OPEN' routine.
e.g. 2430 IF J<>7 THEN PRINT "I can't open the "RI$:RETURN
2440 PRINT"Alright, so what?":RETURN

Line 2430 checks that the object is a box (i.e. the 7th object in the list) and 2440 gives a response to the command 'OPEN BOX'.

If the box is 'valuable' i.e. adds to the score, then:
In line 1460 change the '6' to a '7' to include the box (remember, the box is now the 7th object).
Alter lines 1490 — 1510 to update the maximum number of points possible to '80'.

ADDING A LOCATION

In line 240 increment L by one, to L = 22.
Append new location's description to data list.
e.g. Create line 2041.
2041 DATA "on a vast, red plain. →
→
Some obvious exits: EAST."

And if you get there by 'GO WEST' from location one then alter location one to read:
1840 DATA "at a plateau near a cliff. A rocky → path leads south. →
→
Some obvious exits: SOUTH. WEST."

Now to edit the program to handle 'GO WEST' from location one and 'GO EAST' from location twenty two you will need the following information:

LINE NO.	DIRECTION HANDLED
610-620	NORTHWEST
640-650	NORTHEAST
670-680	SOUTHWEST
700-710	SOUTHEAST
730-800	NORTH
820-880	SOUTH
900-930	WEST
950-980	EAST

```
1000-1010  UP
1030-1040  DOWN
```

e.g. In the 'WEST' routine, create line:

```
925 IF LO=1 THEN LO=22
```

and in the 'EAST' routine, create line:

```
975 IF LO=22 THEN LO=1
```

Now, with some thought, a full 16K custom-made adventure can be written from this 'skeleton' adventure.

The SAVE & LOAD routines in the program were written for disk based micros. Owners of tape bases systems will need to make the following modifications:

DELETE LINES 1620-1650 AND 1700-1730.
NOW INSERT THESE LINES . . .

```
      ** SAVE ROUTINE **
1620  C$="":FOR I9=1 TO OB:
      C$=C$+STR$( B(19) )+"/":
      NEXT I9
1630  PRINT #-1, C$,LO
1640  RETURN

      ** LOAD ROUTINE **
1700  INPUT #-1, C$,LO: IN=O:
      D$=""
1710  FOR I9=1 TO OB
1720  IN=IN+1: M$=MID$ (C$, IN,
      1): D$=D$+M$
1730  IF M$="/" THEN D$=LEFT$
      (D$,LEN(D$)-1):
      B(I9)=VAL(D$): D$="":GOTO
      1740 ELSE 1720
1740  NEXT I9
1750  RETURN
```

BATTLESHIPS (VZED 8K)

This is the old board game of Battleships and cruisers. The screen is divided into a 9 x 9 grid. The computer 'hides' a total of 10 ships at random around this grid. There are four types of ships — 1 Battleship which occupies four adjacent squares, two Cruisers which occupy three adjacent squares each, three destroyers which occupy two adjacent squares each and four submarines occupying yes, you've got it, one square each.

You must enter the coordinates of a square in the grid, at which time the computer prints either a letter in that square, denoting the type of vessel hit, or will print an asterisk if the square is empty. The object of the game is to sink all the vessels with the least possible number of shots. Good hunting!

JUNIOR MATHS (VZED 8K)

This program tests the four basic mathematical functions: Addition, Division, Subtraction and Multiplication. Whilst not an educational program in the strictest sense, it does serve to reinforce lessons already learnt. You are first asked to choose the type of problem after which a graphics screen is presented with an area for the questions and answers and a representation of a persons head with a non-commital expression and some ominous blue water at the bottom. 10 questions are presented one at a time. A correct answer is rewarded by a smile and some uplifting music whilst an incorrect answer causes a frown and depressing music. In this event, the correct answer is also displayed. When the ten questions have been presented, your score and percentage correct are shown.

Now comes the odd bit which may cause our mailbags to bulge with irate letters from outraged child psychologists. In the original version, the author "punished" an imperfect score by raising the water level until it covered the head. He soon found that children using it would deliberately enter incorrect answers just to see this happen. So he reversed the procedure. Now to submerge the hapless head, one must get a perfect score! By the way, the level of difficulty is appropriate to children aged from 9-11.

DISK DIRECTORY PROGRAM (48K/MOD III DISK)

by Ross Smith

REQUIREMENT TO RUN PROGRAM

A 32K or 48K TRS-80 Model III with at least one disk drive. A second drive simplifies the entering of data. A printer is optional. The program was written to be used with TRSDOS 1.3 and will only operate under other operating systems if lines 10 to 30 are modified. These lines use a call to a TRSDOS I/O call ($RAMDIR — 4290H) which is documented in the TRSDOS owner's manual.

DESCRIPTION OF PROGRAM

This program was written to enable the user to keep track of his disk programs. It will maintain a catalog of the name of the program, the extension and the name of the diskette on which the program is stored. The program has been automated as far as possible including the use of INKEY$. The only data that the user needs to enter is the program's name, as the other relevant data is automatically read off the diskette by a machine language subroutine.

The data is stored as linked lists in such a way that all three lists of data can be sorted simultaneously. The data can then be stored in its sorted form on diskette. Thus, although the actual sort can take several minutes, it only needs to be carried out once after new data has been entered into the file.

The program protects enough memory to hold a short machine language program as well as a full diskette directory when it is read from a disk by the TRSDOS I/O call $RAMDIR. As this is done from within the program there is no need to remember to set the memory size before using it.

Several options have been included in this program to allow maximum flexibility and ease of use. The following summarises these options:

(1) ADDING A DISKETTE — Lines 1000 to 1990

This is the fundamental part of the program and allows the contents of up to 100 disks (up to 30 for a 32K machine) to be stored in memory. A total of 700 (300) programs can be stored at a time. After inputting the diskette's name the user is required to put the diskette in the appropriate drive and press /ENTER/. The directory is then automatically read into memory using a machine language program stored in high memory which calls a TRSDOS I/O call. The call ($RAMDIR — 4290H) is clearly documented in the TRSDOS owner's manual. The name of each program on the diskette, its extension and the name of the diskette are stored as a linked list in array D(2,M). The linking occurs through array T(2,M) in such a way that all three lists of information can be sorted at the same time. The diskette name is also added to a separate array A(N) for later use. Before returning to the main menu this array is sorted using Disk BASIC's machine language sort CMD"O". A diskette containing Disk BASIC must be in Drive 0 when this occurs. Thus when using this program on a single drive machine ensure that a diskette containing Disk BASIC is in the drive before hitting /ENTER/ to return to the main menu.

(2) DELETING A DISKETTE — Lines 2000 to 2990

Since the data is stored as linked lists, this routine cannot simply clear the appropriate entries in the relevant array. Instead, a graphic symbol is inserted into the appropriate elements of the arrays which are then sorted. The graphic symbol is thus moved to the end of each of the three columns of the array and can be cleared. As is mentioned below this sort can take a considerable time depending on the number of elements in the array.

(3) UPDATING A DISKETTE — Lines 3000 to 3990

This part of the program uses the above two subroutines to first remove a diskette and then enter the updated version into memory. As with the previous routine this one may take considerable time due to the need to sort the data before deleting the old information.

(4) LISTING DATA — Lines 4000 to 4990

This subroutine allows the data to be listed to the video display. If the printer option is engaged (see below) the data is also sent to a printer. Four options are available. The first three list all the stored data. They differ only in which category is listed first (in alphabetical order if the list has been sorted). The fourth option lists only the diskette names. This option can be used to quickly see which names have already been used.

(5) SORTED DATA — Lines 5000 to 5990

This routine allows the data to be sorted by program name, program extension, diskette name or all three. The data is stored in array D(2,M) as three linked lists using array T(2,M) to maintain the links. The data in each of the three columns thus can be in-

dividually sorted with the appropriate links between the data being maintained by array T(2,M). Although the program uses a Shell-Metzner sort to increase the speed of the sort, a sort on a large number of elements may take several minutes as three separate sorts may be involved. For example a sort on all three fields of 200 programs will take approximately 6.0 minutes. Note that this routine is also called whenever a diskette is updated or deleted.

(6) SEARCHING FOR DATA — Lines 6000 to 6990

This is a very versatile routine which allows a search to be carried out on one, two or all three fields of data. The search may be for an exact match (exclusive) or for a match with only part of the data (inclusive). For example an exclusive search for DOS in the program name field would only return a match if a program named DOS was found. On the other hand an inclusive search would also find TRSDOS and DOSPLUS if they were present. Up to six separate strings can be searched for in any of the three fields simultaneously. The data will be sent to a printer as well as the video if the printer option has been engaged.

(7) PRINTING DATA — Lines 7000 to 7990

This is a short subroutine which turns a print flag on (Z = 1) or off (Z = 0). Initially the flag is off (Z is set to 0 in line 40). This flag determines whether the output from the LIST and FIND routines is sent to the printer as well as the video screen. Note that once engaged this option will continue to direct output to both the printer and screen until it is disengaged. It is therefore necessary to call this routine after getting a printout if further printouts are not required.

(8) WRITING DATA TO DISK — Lines 8000 to 8990

This section of the program writes the stored data to a diskette in Drive 0. After being called the routine asks whether to write to disk or not. This is the user's last chance to change his mind. Answering the question with an N will return you to the main menu. The program uses a filename of DISKDIR/DAT for the data file.

MODIFICATIONS

The program as written is for a dual drive machine. It can be modified for a single drive by changing line 12 to read DISK% = 0. This means that extra disk swapping may be required. Note that the main diskette must contain Disk BASIC. When using a single drive any diskette not containing Disk BASIC must be replaced with a disk which contains BASIC before returning to the main menu after entering a new disk directory,

The program has been written for a 48K machine which accommodates 100 disks containing 700 programs. The program, which only takes up 6500 bytes, can be modified for a 32K machine by changing the following lines:

Finally all data is presently stored on Drive 0 in a file named DISKDIR/DAT. To change this it is necessary to modify lines 110 and 8000.

VARIABLES

INTEGER I-Q and S-Z
STRING A-H

A(N)	Diskette names
B(2,5)	Strings for search routine
C(2)	Field titles for list and print routines
D(2,M)	Program names (0), Extensions (1) and Diskette names (2)
S(2)	No. of strings to be searched for in each field
T(2,M)	Links between Program names, extensions and Diskette names
A1	Program name
A2	Program extension
A4	List of 1st letters of allowable inputs
A5 to A7	Field names for list and print routines
B	INKEY$ input
C	Program name input
J	INKEY$ input converted to numeric
J2	Delete only or update diskette flag
J3	Flag to check if disk to be removed is on file
L1	Used when retrieving data from the directory = 22 if previous program name has an extension = 23 if previous program name does not have an extension
M	Maximum no. of programs
M1	Actual no. of programs
N	Maximum no. of diskettes
N1	Actual no. of diskettes
Z	Printer on/off flag
Z1	Type of search flag (extensive/intensive)

I, I1, I2, I3, J, K, K2, K3 — Loop variables
Others Temporary variables

FNP (L,P) — Calculates video screen location of Position P on Line L

PEEKS AND POKES

14400 check if ENTER is pressed
16412 Non-blinking cursor
16419 Sets cursor character
16427 Sets maximum printer length
16561/62
Top of memory
16916 Screen scroll protect
17425/26
Top of memory

SUGGESTED IMPROVEMENTS

The single greatest improvement that could be made to this program would be to increase the speed of the sort routine. This would probably mean going to a machine language sort since the sort used by the program is very efficient. The Disk BASIC sort cannot be used as the three sets of information are linked through a separate array. A specialised routine would be needed. A significant factor in the sort time for larger sorts is BASIC's garbage collection routine. Any method of reducing this would greatly reduce the time for larger sorts.

NOTES

It should be noted that the program will occasionally stop while outputting a diskette directory to the screen. During these periods all control of the keyboard will be lost. This is due to BASIC's garbage collection routine and the only thing to do is to wait until control is regained. The period of loss of control can be quite long as the number of stored programs increases.

INPUT/OUTPUT

In this column we answer Readers' letters. We also encourage other Readers who have experience of the problems reported to write in with their solutions. We are happy to receive requests for help in solving Adventure games etc. but do not believe in giving direct answers, that would just spoil the game for the Reader concerned and many others. We will give hints and cryptic clues (if we have managed to solve the game ourselves!!)

HOUSEHOLD ACCOUNTS UNDER NEWDOS 80

FROM: Rosemary Low
Wavell Heights, Qld.

Many thanks indeed for the free software pack. I was particularly interested in the Home Accounting Software Package and on trying to run it on my Model 1 first up found one not so obvious 'bug' in the program for which I received the error message "Syntax error in line 8"!!! — but after listing out the program discovered the actual problem lay in line 250 and that in fact there was no line 8!!! After trying to edit line 250 I discovered that it actually extended beyond 250 characters and so I had to cut out some of the unnecessary spaces. Line 250 lists the main menu of the accounting program (options 1 to 8). After making that correction the program worked fine. So I feel if others are having trouble debugging it this may help to put them on the right track. Actually in the end I had to retype the whole of line 250 again which reads:

```
250 P=0:GOSUB230:PRINT@220:"MENU
1 = KEYBOARD INPUT  5 = SAVE DATA
2 = LOAD DATA       6 = PRINT JOURNALS
3 = READ " ;Z1$;"   7 = LINEPRINTER UTILITY
4 = EDIT " ;Z1$;"   8 = LEDGER ";Z2$;" S";
```

260' (option 8 is put into a less spacious line 250 so that I could more easily line up the 8 options). — but line 260 could be left as it was and line 250 ended just prior to where option 8 should begin.

To make the program work more satisfactorily on Newdos-80 I amended the following lines for the file save to disk then file load from disk. I used "MU" files as they are meant on Newdos-80 to replace sequential files under the TRSDOS setup:

```
1510 IFSF=2THENOPEN"I", 1,NM$, "MU"
```

1530 IFSF=2THENGET 1,,, W;: FORI=1TOW:GET 1,,,A$(I);:NEXT:CLOSE 'LOAD FROM DISK

1410 IFSF=2THEN OPEN"O", 1,NM$,"MU"

1430 IFSF=2THENPUT 1,,,W;:FORI=1TOW:PUT 1,,,A$(I);:NEXT:CLOSE 'SAVE TO DISK

The only problem with "MU" files is that they cannot be updated as can "MI", "FI" or "MF". "MU" files also have no specific record length and being sequential files can therefore take up less space than random files. I find that under Newdos the PRINT #1 and INPUT #1 do not actually save any file as they hadn't been incorporated into the Newdos filing system and therefore this has to be allowed for in dealing with Newdos files. I do hope this will put some other Newdos users on the right track too. Thanking you for your help and co-operation.

(Thank you for this contribution Rosemary — Ed.)

DATABASE REVISITED

FROM: Graeme Moad — Windsor Vic.

I am writing to let you know that I have been found (by Jim Campbell, see: Input/Output January 1984), and to let people know of a couple of bugs in my database program (published in the January 1982 issue of MICRO-80) which he brought to my attention.

The problem is that the program does not store data placed in integer fields properly. This can be corrected by modifying the last part of line 310 of the program which currently reads:

:POKE VARPTR (DU(I)), 48: NEXT TO:

:POKE VARPTR (DU(I)), F(1,I) :NEXT

In addition line 380 of the program should be changed from:

380 ON M GOSUB 1150,1260 :IF IE<>0 THEN 360 ELSE 390 TO:

380 ON M GOSUB 1150,1260 :IF IE<>0 THEN GOSUB 120 :RUN

To avoid a possible "redimensioned array" error.

If readers of MICRO-80 have found other bugs in this program please let MICRO-80 know so that appropriate corrections can be published. As readers will no doubt put (have put) the program to many uses which I have not anticipated (such as using integer fields) other bugs may still be lying in wait for the unwary.

Given sufficient reader interest, I would also be willing to supply MICRO-80 with a substantially revised and commented version of the program which (a) adds a number of feature and (b) would enable interested readers to more readily make their own modifications to the program. The version published was packed (maximum statements per line, no comments) so as to minimize the amount of memory taken up and allow the maximum room for the database.

(There has been considerable interest in this program Graeme. Please send in your revised version. — Ed).

```
        **** Dogfight ****

         COLOUR COMPUTER


i0 '    **********************
        *  STEPHEN GIBBONS   *
        * 34 THE COMENARRA   *
        *  PKY, THORNLEIGH   *
        *    N.S.W. 2120     *
        **********************
20 CLS
30 PRINT:PRINT"              dogfi
ght"
35 SOUND 89,1:SOUND1251:SOUND147
,1:SOUND176,7
40 PRINT:PRINT
50 PRINT"         BY S.GIBBONS '8
3"
55 SOUND218,2:SOUND218,9
60 PRINT
70 INPUT"LEFT PLAYER'S NAME";LP$
:PRINT:INPUT"RIGHT PLAYER'S NAME
";RP$
80 PRINT
90 PRINT"DO YOU HAVE JOYSTICKS (
Y/N)"
100 AA$=INKEY$:IF AA$="" THEN 10
0
110 IF AA$="Y" THEN 120 ELSE 140

120 PRINT:PRINT"PLUG IN JOYSTICK
S.  HIT RETURN"
130 I$=INKEY$:IFI$=""THEN130
140 PRINT:PRINT"DO YOU NEED INST
RUCTIONS (Y/N)?";
150 I$=INKEY$:IFI$="" THEN 150
160 IF I$="Y" THENGOSUB 2060
170 CLS0
180 X=RND(31)+15:A=RND(31)+15:Y=
RND(15)+7:B=RND(15)+7
190 M=RND(7)+23:FORN=M TO31:SET(
0,N,5):NEXT
200 FORN=1TO63
210 ND=RND(2):ON ND GOSUB260,280

220 SET(N,M,5):SET(N-1,M,5):NEXT

230 FORN=M TO31:SET(63,N,5):NEXT

240 FORN=0TO63:SET(N,31,5):NEXT
250 GOTO300
260 IFM<23THENM=M+1:RETURN
270 M=M-1:RETURN
280 IFM>30THENM=M-1:RETURN
290 M=M+1:RETURN
300 I$=INKEY$
310 IFX=A AND Y=B THEN1420
320 IF I$="^" THEN 1510
330 IF AA$<>"Y" THEN350
340 IF PEEK(65280)=125 OR PEEK(6
5280)=253 THEN 1510
350 IFP3=1THEN430
360 IF I$="Q" THEN AD=AD-1:IF AD
<1 THEN AD=8
370 IF I$="W" THEN AD=AD+1:IF AD
>8 THEN AD=1
380 IF AA$<>"Y"THEN 410
390 IF JOYSTK(2)<10 THEN AD=AD-1
:IFAD<1 THEN AD=8
400 IFJOYSTK(2)>53 THENAD=AD+1:I
FAD>8THENAD=1
410 IF I$="@" THEN GOSUB 1870
420 IF AA$<>"Y" THEN 440
430 IF PEEK(65280)=126 OR PEEK(6
5280)=254 THEN GOSUB 1870
440 IFP2<>1THEN490
450 IFP2=1 THENW=RND(2):ON W GOT
O 460,470
460 BD=6:GOTO480
470 BD=4
480 IFP2=1THEN550
490 IF I$=CHR$(8) THEN BD=BD-1:I
FBD<1THENBD=8
500 IF I$=CHR$(9) THEN BD=BD+1:I
F BD>8 THEN BD=1
510 IF AA$<>"Y" THEN 540
520 IF JOYSTK(0)<10 THENBD=BD-1:
IF BD<1 THEN BD=8
530 IFJOYSTK(0)>53THENBD=BD+1:IF
BD>8THENBD=1
540 IFP3=1THENQ=RND(2):ON Q GOTO
560,570
550 GOTO580
560 AD=4:GOTO580
570 AD=6
580 ON AD GOSUB 670,690,710,730,
750,770,790,810
590 IFX>61THENRESET(61,Y):RESET(
61,Y-1):RESET(61,Y+1):X=3
600 IFX<3THENRESET(X+1,Y):RESET(
X+1,Y+1):RESET(X+1,Y-1):X=61
610 IFY<2THENY=2
620 SET(X,Y,3)
630 IFP3=1THEN1300
640 RESET(X-1,Y):RESET(X+1,Y):RE
SET(X,Y+1):RESET(X,Y-1):RESET(X-
1,Y-1):RESET(X+1,Y-1):RESET(X-1,
Y+1):RESET(X+1,Y+1)
650 GOTO1300
660 Y=Y-1:RETURN
```

```
670 IFPOINT(X,Y-2)=5THEN830
680 Y=Y-1:RETURN
690 IFPOINT(X+2,Y-2)=5THEN830
700 X=X+1:Y=Y-1:RETURN
710 IFPOINT(X+2,Y+2)=5THEN830
720 X=X+1:RETURN
730 IFPOINT(X+1,Y+2)=5THEN830
740 X=X+1:Y=Y+1:RETURN
750 IFPOINT(X,Y+2)=5THEN830
760 Y=Y+1:RETURN
770 IFPOINT(X-1,Y+2)=5THEN830
780 X=X-1:Y=Y+1:RETURN
790 IFPOINT(X+2,Y+2)=5THEN830
800 X=X-1:RETURN
810 IFPOINT(X-2,Y-2)=5THEN830
820 X=X-1:Y=Y-1:RETURN
830 FORZ=1TO5:SOUNDRND(5)+250,1:
NEXT
835 PRINT@10,RP$;" WINS";
840 FORZ=1TO25
850 P3=0
860 C=RND(8)
870 SET(X,Y+2,C):SET(X-1,Y-1,C):
SET(X+1,Y-2,C)
880 SET(X+2,Y-4,C):SET(X-2,Y-5,C
)
890 RESET(X,Y)
900 SET(X,Y+1,C)
910 NEXT
920 SC=SC+1:GOTO 1191
930 RESET(X,Y+1):RESET(X+1,Y-2):
RESET(X-1,Y-1):RESET(X,Y-3):RESE
T(X-2,Y-5)
940 AD=0:BD=0:GOTO140
950 GOTO 140
960 IFPOINT(A,B-2)=5THEN1120
970 B=B-1:RETURN
980 IFPOINT(A+2,B-2)=5THEN1120
990 A=A+1:B=B-1:RETURN
1000 IFPOINT(A+2,B+2)=5THEN1120
1010 A=A+1:RETURN
1020 IFPOINT(A+1,B+2)=5THEN1120
1030 A=A+1:B=B+1:RETURN
1040 IFPOINT(A,B+2)=5THEN1120
1050 B=B+1:RETURN
1060 IF POINT(A-2,B)=5 OR POINT(
A,B+2)=5 THEN1120
1070 A=A-1:B=B+1:RETURN
1080 IFPOINT(A-2,B+2)=5THEN1120
1090 A=A-1:RETURN
1100 IFPOINT(A-2,B-2)=5THEN1120
1110 A=A-1:B=B-1:RETURN
1120 FORZ=1TO5:SOUNDRND(5)+250,1
:NEXT
1125 PRINT@10,LP$;" WINS";
1130 FORZ=1TO25
1140 C=RND(8)
1150 SET(A,B+1,U):SET(A+1,B-2,C)
:SET(A-2,B-3,C):SET(A+3,B-4,C)
1160 SET(A,B-1,C)
1170 SET(A-1,B-5,C)
1180 RESET(A,B)
1190 NEXT:P2=0:SD=SD+1
1191 SOUND133,7:SOUND133,7:SOUND
133,2:SOUND133,7
1192 SOUND153,7:SOUND 147,2:SOUN
D 147,7
1193 SOUND133,2:SOUND 133,7:SOUN
D125,2:SOUND133,9
1220 CLS:PRINT"                scor
es"
1230 PRINT:PRINT:PRINT
1240 PRINTTAB(5)LP$;TAB(20)SD:PR
INT:PRINTTAB(5)RP$;TAB(20)SC
1250 FORD=1TO1000:NEXT
1260 IF SC>9 OR SD>9 THEN 1950
1270 AD=0:BD=0
1280 CLS0
1290 GOTO 170
1300 '
1310 ON BD GOSUB 960,980,1000,10
20,1040,1060,1080,1100
1320 IFA>61THENRESET(61,B):RESET
(61,B-1):RESET(61,B+1):A=3
1330 IFA<3THENRESET(A+1,B):RESET
(A+1,B+1):RESET(A+1,B-1):A=61
1340 IFB<3THENC=RND(2):ON C GOTO
1370,1360
1350 GOTO1380
1360 B=3:BD=7:GOTO1380
1370 B=3:BD=3:GOTO1380
1380 SET(A,B,4)
1390 IFP2=1THEN300
1400 RESET(A+1,B):RESET(A-1,B):R
ESET(A,B-1):RESET(A,B+1):RESET(A
+1,B-1):RESET(A-1,B-1):RESET(A+1
,B+1):RESET(A-1,B+1)
1410 GOTO300
1420 FORZ=1TO7:SOUNDRND(5)+250,1
:NEXT
1425 PRINT@9,"YOU BOTH LOSE";
1430 FORZ=1TO25
1440 C=RND(8)
1450 SET(X-2,Y+2,C):SET(X+1,Y+4,
C):SET(X+2,Y-3,C):SET(X-1,Y-2,C)

1460 SET(X+1,Y+2,C)
1470 SET(X+3,Y,C)
1480 NEXT
1490 SC=SC+1:SD=SD+1
1500 AD=0:BD=0:GOTO 1191
1510 M=X:N=Y
1520 K=0
1530 ON AD GOSUB 1560,1580,1600,
1620,1640,1660,1680,1700
1540 GOTO1720
1550 RETURN
1560 IFPOINT(M,N-2)=5THEN1820
1570 N=N-1:RETURN
1580 IFPOINT(M+2,N-2)=5THEN1820
1590 N=N-1:M=M+1:RETURN
1600 IFPOINT(M+2,N)=5THEN1820
1610 M=M+1:RETURN
1620 IFPOINT(M+2,N+2)=5THEN1820
1630 M=M+1:N=N+1:RETURN
1640 IFPOINT(M,N+2)=5THEN1820
1650 N=N+1:RETURN
1660 IFPOINT(M-2,N+2)=5THEN1820
1670 N=N+1:M=M-1:RETURN
1680 IFPOINT(M-2,N)=5THEN1820
1690 M=M-1:RETURN
1700 IFPOINT(M-2,N-2)=5THEN1820
1710 M=M-1:N=N-1:RETURN
1720 IFPOINT(M,N-1)=4 OR POINT(M
,N+1)=4 OR POINT(M+1,N)=4 OR POI
NT(M-1,N)=4 THENP2=1
1730 IF POINT(M-1,N-1)=4 OR POIN
T(M-1,N+1)=4 OR POINT(M+1,N-1)=4
 OR POINT(M+1,N+1)=4 THENP2=1
1740 K=K+1:IFK>5THENK=0:GOTO1820

1750 IFM<2 OR M>60 ORN<2THENK=0:
GOTO1820
1760 SOUND243,1
1770 SET(M,N,2):SET(X,Y,3):SET(,


B,4)
1780 RESET(M+1,N):RESET(M-1,N):R
ESET(M,N+1):RESET(M,N-1)
1790 RESET(M-1,N+1):RESET(M+1,N+
1):RESET(M-1,N-1):RESET(M+1,N-1)

1800 IFGH=1THENGOTO1890
1810 GOTO1530
1820 RESET(M,N)
1830 RESET(M-1,N):RESET(M+1,N):R
ESET(M,N-1):RESET(M,N+1)
1840 RESET(M-1,NISA-1):RESET(M-1
,N+1):RESET(M+1,N-1):RESET(M+1,N
+1)
1850 GH=0
1860 GOTO300
1870 M=A:N=B
1880 K=0:GH=1
1890 ON BD GOSUB 1560,1580,1600,
1620,1640,1660,1680,1700
1900 GOTO1920
1910 RETURN
1920 IFPOINT(M,N-1)=3 OR POINT(M
,N+1)=3 OR POINT(M+1,N)=3 OR POI
NT(M-1,N)=3 THENP3=1
1930 IFPOINT(M-1,N-1)=3 OR POINT
(M-1,N+1)=3 OR POINT(M+1,N-1)=3
OR POINT(M+1,N+1)=3 THENP3=1
1940 GOTO1740
1950 CLS
1960 PRINT
1970 PRINT"              dogfight"
1980 PRINT:PRINT
```

```
1990 IF SC>SD THEN CH$=RP$:LO$=L
P$ ELSE CH$=LP$:LO$=RP$
2000 PRINTCH$;" IS THE BETTER BA
RRON."
2010 PRINT"BETTER LUCK NEXT TIME
 "$LO$;"."
2011 SOUND89,2:SOUND89,2:SOUND89
,2:SOUND125,2:SOUND89,2
2012 SOUND125,2:SOUND147,2:SOUND
125,2:SOUND147,2:SOUND176,7
2013 FORZZ=1TO100:NEXT:SOUND 176
,2:SOUND176,7
2020 PRINT:PRINT
2030 PRINT"ANOTHER DOGFIGHT (Y/N
) ?"
2040 I$=INKEY$:IFI$=""THEN2040
2050 IFI$="Y" THEN RUN ELSE CLS:
END
2060 CLS:PRINT"            dogfig
ht"
2070 PRINT
2080 PRINT"    DOGFIGHT IS A GAM
E FOR TWO  PLAYERS.  THE RULES A
RE SIMPLE. SHOOT DOWN YOUR OPPON
ENT BEFORE HE SHOOTS YOU DOWN."
2090 PRINT"    THE WINNER IS THE
 PILOT WHO WINS TEN ROUNDS FIRST
."
2100 PRINT"    IF YOU CRASH INTO
 THE       MOUNTAINS YOU WILL EX
PLODE.     IF YOU CRASH INTO YOU
R OPPONENT, YOU BOTH WILL EXPLOD
E."
2110 PRINT:PRINT"HIT ANY KEY TO
CONTINUE."
2120 I$=INKEY$:IFI$="" THEN 2120
 ELSE CLS
2130 PRINT"            dogfight"

2140 PRINT
2150 PRINTLP$;"'S CONTROLS ARE:"
2160 PRINT"<Q>    -TO ROTATE ANTI
CLOCKWISE"
2170 PRINT"<W>    -TO ROTATE CLOC
KWISE      <^>    -TO FIRE MACHIN
E GUNS"
2180 PRINTRP$;"'S CONTROLS ARE:"

2190 PRINT"<LEFT ARROW>     -ROTA
TE LEFT    <RIGHT ARROW>    -ROTA
TE RIGHT"
2200 PRINT"<@>              -TO F
IRE"
2210 PRINT"IF YOU HAVE JOYSTICKS
, PUSH     LEVER LEFT TO ROTATE
LEFT, RIGHTTO ROTATE RIGHT AND F
IRE BUTTON TO FIRE."
2220 PRINT:PRINT"HIT ANY KEY TO
CONTINUE";
2230 I$=INKEY$:IF I$="" THEN 223
0 ELSE RETURN
```

**** Model III/Disk Disk Directory Recorder ****

TRS-80/SYSTEM-80

```
1 ' DISKETTE DIRECTORY
2 ' WRITTEN BY ROSS JAMES SMITH
3 ' VERSION 1.3
4 ' COPYRIGHT DECEMBER 1982
5 '68 BLAKESLEY ROAD,
   SOUTH HURSTVILLE,
   N.S.W.      2221.
6 '
7 'FOR SINGLE DRIVE SYSTEMS CHANGE :-
   LINE 12 TO DISK%=0
8 '
9 CMD"B","OFF":POKE16419,95:POKE16412,1:POKE16427,134
10 POKE16561,220:POKE16562,246:POKE17425,224:POKE17426,246:CLEAR
18000:DEFUSR1=&HF6E0:DEFINTI-Q,S-Z:DEFSTRA-H,R:DEFFNP(L,P)=(L-1)
*64+P
11 PRINTCHR$(22)
12 DISK%=1
20 K=&HF6E0:FORI=KTOK+9:READJ:POKEI,J:NEXTI
30 DATA33,235,246,1,0,1,205,144,66,201
35 IFDISK%=0THENPOKEK+5,0
40 Z1=0:Z=0:M=700:N=100:N1=0:M1=0
50 DIMD(2,M),T(2,M),B(2,5),A(N),C(2),S(2)
60 CLS:PRINT@FNP(8,21),"NEW FILE (Y/N) ?";:GOSUB12000:IFB<>"Y"AN
DB<>"N"THEN60ELSEPRINT" ";B;
70 IFB="Y"THEN200
110 OPEN"I",1,"DISKDIR/DAT:0":INPUT#1,M1,N1:FORJ=0TO2:FORI=1TOM1
:INPUT#1,D(J,I),T(J,I):NEXTI:NEXTJ:FORI=1TON1:INPUT#1,A(I):NEXTI
:CLOSE1
200 CLS:PRINT@FNP(6,21),"No. of Diskettes = ";N1;:PRINT@FNP(8,21
),"No. of Programs  = ";M1;:GOSUB10000
210 C(0)="PROGRAM  ":C(1)="EXTENSION":C(2)="DISKETTE "
220 A4="ADULSFPWE"
300 CLS:PRINT@FNP(2,24),"DISKETTE INDEX";
                                D - DELETE DISKETTE DIR
                                U - UPDATE DISKETTE DIR
                                L - LIST DATA
                                S - SORT DATA
                                F - FIND DATA"
320 PRINT@FNP(11,20),"P - PRINT DATA
                                W - WRITE DATA TO DISK
                                E - END PROGRAM""
330 GOSUB12000:FORI=1TO9:IFB=MID$(A4,I,1)THENONIGOSUB1000,2000,3
000,4000,5000,6000,7000,8000,9000:GOTO300ELSENEXTI:GOTO330
999 '  *****  DISKETTE ADD SUBROUTINE  *****
1000 CLS:PRINT@FNP(8,18),"INPUT DISKETTE NAME";:INPUTC:FORI=1TON
1:IFA(I)=CTHENPRINT@FNP(12,17+LEN(C)/2),"DISKETTE ALREADY ON FIL
E";:GOSUB10000:GOTO1990ELSENEXTI:IFN1>=NTHENPRINT@FNP(12,25),"ME
MORY FULL";:GOSUB10000:GOTO1990
1010 PRINT@FNP(10,17+LEN(C)/2),"PUT DISKETTE IN DRIVE";DISK%;:GO
SUB10000:I=&HF6EB:CLS:PRINT@FNP(1,23),C;" DIRECTORY"
1020 X=USR1(0):IFPEEK(I)=43THENPRINT@FNP(8,27),"NO ENTRIES":GOSU
B10000:GOTO1990ELSEN1=N1+1:A(N1)=C
1030 IFM1>=MTHENPRINT@FNP(8,27),"MEMORY FULL";:GOSUB10000:GOTO19
90ELSEA="":FORJ=0TO14:L=PEEK(I+J):IFL<>32ANDL<>58THENA=A+CHR$(L)
:NEXTJ
1040 M1=M1+1:A1="":A2="":FORI2=1TOLEN(A):IFMID$(A,I2,1)<>"/"THEN
A1=A1+MID$(A,I2,1):NEXTI2:GOTO1055
1050 FORI3=I2+1TOLEN(A):IFMID$(A,I3,1)=":"THEN1055ELSEA2=A2+MID$
(A,I3,1):NEXTI3
1055 A3=A1+A2:FORI2=1TOLEN(A3):IFASC(MID$(A3,I2,1))>32THENNEXTI2
:GOTO1060:ELSEA1="":M1=M1-1:GOTO1065
1060 D(0,M1)=A1:D(1,M1)=A2:D(2,M1)=C:T(0,M1)=M1:T(1,M1)=M1:T(2,M
1)=M1:IFA2=""THEND(1,M1)="..."
1065 IFLEN(A2)>0THENL1=22ELSEL1=21
1070 IFPEEK(I+L1)<>43THENI=I+L1:IFA1<>""THENPRINTA1;"/";A2,:GOTO
1030:ELSEGOTO1030
1080 IFA1<>""THENPRINTA1;"/";A2
1085 GOSUB10000
1090 CMD"O",N1,A(1)
1100 CLS:PRINT@FNP(7,21),"No. of Diskettes = ";N1;
1110 PRINT@FNP(9,21),"No. of Programs  = ";M1;:GOSUB10000
```

```
1990 RETURN
1999 '  *****  DISKETTE DELETE SUBROUTINE  *****
2000 J2=0
2010 B=CHR$(191):J3=0:CLS:PRINT@FNP(8,18),"INPUT DISKETTE NAME";
:INPUTC
2015 PRINT@FNP(10,23),"REMOVING DISKETTE";
2020 FORI=1TOM1:IFD(2,I)=CTHEND(2,I)=B:J5=T(2,I):D(0,J5)=B:J5=T(
0,J5):D(1,J5)=B:J3=J3+1
2030 NEXTI:IFJ3=0THENPRINT@FNP(10,22),"DISKETTE NOT IN FILE":GOS
UB10000:GOTO2990
2040 FORI=1TON1:IFA(I)=CTHENA(I)=B:GOTO2050ELSENEXTI
2050 CMD"O",N1,A(1):GOSUB5300:N1=N1-1:M1=M1-J3
2060 IFJ2=1THEN2990ELSECLS:PRINT@FNP(7,20),"Diskettes Remaining
= ";N1;
2070 PRINT@FNP(9,20),"Programs Remaining = ";M1;:GOSUB10000
2990 RETURN
2999 '  ***** DISKETTE UPDATE SUBROUTINE  *****
3000 J2=1:GOSUB2010:IFJ3=0THEN3990ELSECLS:GOSUB1010
3990 RETURN
3999 '  *****  LIST SUBROUTINE  *****
4000 CLS:PRINT@FNP(4,26),"LIST DATA BY
                    1 - PROGRAM NAME
                    2 - PROGRAM EXTENSION
                    3 - DISKETTE NAME
                    4 - LIST DISKETTES ONLY"
4010 GOSUB10100:IFJ<1ORJ>4THEN4010ELSEJ=J-1:IFJ=3THEN4500
4020 CLS:GOSUB11100:POKE16916,2
4030 FORL=1TOM1STEP13:FORK=0TO12:IFL+K>M1THENNEXTK:GOTO4050
4040 I=L+K:GOSUB11000:NEXTK
4050 GOSUB10000:CLS:NEXTL:POKE16916,0
4060 GOTO1100
4500 CLS:PRINT@FNP(1,28),"DISKETTE":PRINT:POKE16916,2
4505 IFZ=1THENLPRINTAB(28)"DISKETTE":LPRINT
4510 FORI=1TON1STEP13:FORK=0TO12:IFI+K>N1THENNEXTK:GOTO4530
4520 A="                              %       %":PRINTUSINGA;A(I+K)
4525 IFZ=1THENLPRINTUSINGA;A(I+K)
4526 NEXTK
4530 GOSUB10000:CLS:NEXTI:POKE16916,0
4990 RETURN
4999 GOTO4990
5000 CLS:PRINT@FNP(4,24),"SORTING ROUTINE
                    1 - PROGRAM NAMES
                    2 - PROGRAM EXTENTIONS
                    3 - DISKETTE NAMES
                    4 - ALL OF THE ABOVE";
5010 GOSUB10100:J=J-1:IFJ<0ORJ>3THEN5010ELSEPRINT@FNP(11,28),"SO
RTING";:IFJ=3THEN5300
5020 GOSUB5030:GOTO5990
5030 L=M1
5040 L=INT(L/3)+1
5060 FORK=1TOM1-L
5065 POKE16038,161
5070 IFD(J,K)<=D(J,K+L)THEN5180
5080 B=D(J,K+L):T1=T(J,K+L):T=K
5090 D(J,T+L)=D(J,T):T(J,T+L)=T(J,T):T=T-L
5100 IFT>0THENIFB<D(J,T)THEN5090
5110 D(J,T+L)=B:T(J,T+L)=T1
5180 POKE16038,146:NEXTK
5185 IFL>1THEN5040
5200 FORI=1TOM1:T5=T(J,I):J1=J+1:IFJ1=3THENJ1=0
5210 T6=T(J1,T5):J1=J1+1:IFJ1=3THENJ1=0
5220 T(J1,T6)=I:NEXTI:RETURN
5300 FORJ=0TO2
5310 GOSUB5030
5320 NEXTJ
5990 RETURN
5999 '  *****  SEARCH SUBROUTINE  *****
6000 CLS:FORI=1TO5:FORJ=0TO2:B(J,I)="":NEXTJ:NEXTI:S=0:J7=0:J8=0
:J4=0:PRINT@FNP(1,20),"SEARCH ROUTINE";
6005 PRINT@FNP(2,3),"1. Program Name    2.Extension    3.Diskett
e Name";
6010 FORK=0TO2
6012 PRINT@FNP(3,12),"DO YOU WISH TO SEARCH FIELD";K+1;"(Y/N)";:
GOSUB12000:IFB="Y"THENJ4=J4+1:B(K,0)=CHR$(254)ELSEIFB="N"THENB(K
,0)=CHR$(255)ELSEGOTO6012
6014 NEXTK
6016 PRINT@FNP(3,12),CHR$(30);:PRINT@FNP(3,12),"INCLUSIVE OR EXC
LUSIVE (I/E)";:GOSUB12000:IFB="I"THENZ1=0ELSEIFB="E"THENZ1=1ELSE
GOTO6016
6018 IFJ4=0THEN6990
6020 FORK=0TO2:IFB(K,0)=CHR$(255)THEN6045
6030 PRINT@FNP(5+S,6),"Input Field";K+1;" String";S+1;" (=ENTER=
 to Stop)";:INPUTC:IFC=""THEN6040ELSEB(K,S)=C:C="":S=S+1:IFS<5TH
EN6030
6040 S(K)=S:S=0:PRINT@FNP(3,1),CHR$(31);
6045 NEXTK
6050 FORJ=0TO2:IFB(J,0)=CHR$(255)THEN NEXTJ
6055 CLS:GOSUB11100:POKE16916,2
6060 FORI=1TOM1
6070 J5=I:K=0
6080 IFB(K,0)=CHR$(255)THENK=K+1:IFK<3GOTO6080
6100 FORK1=0TOS(K)-1
6110 IF(Z1=0ANDINSTR(D(K,J5),B(K,K1))=0)OR(Z1=1AND D(K,J5)<>B(K,
K1))THENNEXTK1:GOTO6250
6120 IFJ4=1GOTO6240
6130 IFB(1,0)=CHR$(255)THENJ5=T(K,I):K=K+1:GOTO6200
6150 J5=T(K,I):K=K+1
6160 FORK2=0TOS(K)-1
6170 IF(Z1=0ANDINSTR(D(K,J5),B(K,K2))=0)OR(Z1=1AND D(K,J5)<>B(K,
K2))THENNEXTK2:GOTO6250
6180 IFJ4=2GOTO6240
6200 J5=T(K,J5):K=K+1
6210 FORK3=0TOS(K)-1
6220 IF(Z1=0ANDINSTR(D(K,J5),B(K,K3))=0)OR(Z1=1AND D(K,J5)<>B(K,
K3))THENNEXTK3:GOTO6250
6240 GOSUB11000:J7=J7+1:J8=J8+1:IFJ7=13THENJ7=0:GOSUB10000:CLS
6250 NEXTI:IFJ7=0THENIFJ8=0THENPRINT@FNP(8,26),"NO ENTRIES";:IFZ
=1THENLPRINT:LPRINT:LPRINT:LPRINTTAB(26),"NO ENTRIES":LPRINT
6260 GOSUB10000:POKE16916,0
6990 RETURN
6999 '  *****  PRINT SUBROUTINE  *****
7000 CLS:PRINT@FNP(8,18),"OUTPUT TO PRINTER (Y/N)? ";:GOSUB12000
:IFB<>"Y"ANDB<>"N"THEN7000
7010 IFB="Y"THENZ=1ELSEZ=0
7990 RETURN
7999 '  *****  STORE TO DISK SUBROUTINE  *****
8000 CLS:PRINT@FNP(8,21),"WRITE TO DISK (Y/N)? ";:GOSUB12000:IFB
="N"THEN7990ELSEIFB<>"Y"THEN8000
8010 OPEN"O",2,"DISKDIR/DAT:0":CLS:PRINT@FNP(8,24),"WRITING TO D
ISK";:PRINT#2,M1,N1:FORJ=0TO2:FORI=1TOM1:PRINT#2,D(J,I);",";T(J,
```

```
I);:NEXTI:NEXTJ:FORI=1TON1:PRINT#2,A(I);",";:NEXTI:CLOSE2
8990 RETURN
8999 '  *****  PROGRAM END  *****
9000 CMD"B","ON":NEW:STOP
9500 END
10000 PRINT@FNP(16,19),"PRESS =ENTER= TO CONTINUE";
10010 FORI1=1TO50:IFPEEK(14400)=1THENRETURNELSENEXTI1:PRINT@FNP(
16,25),"        ";:FORI1=1TO50:IFPEEK(14400)=1THENRETURNELSENEXTI
1:PRINT@FNP(16,25),"=ENTER=";:GOTO10010
10100 B=INKEY$:B=""
10110 B=INKEY$:IFB=""THEN10110ELSEJ=VAL(B):RETURN
11000 A="       %       %             %       %             %
 %"
11010 I1=T(J,I):J1=J+1:IFJ1=3THENJ1=0
11020 I2=T(J1,I1):J2=J1+1:IFJ2=3THENJ2=0
11030 PRINTUSINGA;D(J,I),D(J1,I1),D(J2,I2)
11040 IFZ=1THENLPRINTUSINGA;D(J,I),D(J1,I1),D(J2,I2)
11090 RETURN
11100 J1=J:A5=C(J1):J1=J1+1:IFJ1=3THENJ1=0
11110 A6=C(J1):J1=J1+1:IFJ1=3THENJ1=0
11120 A7=C(J1):PRINTTAB(6)A5;TAB(27)A6;TAB(48)A7
11130 PRINT
11140 IFZ=1THENLPRINTTAB(6)A5;TAB(27)A6;TAB(48)A7:LPRINT
11190 RETURN
12000 B=INKEY$:B=""
12010 B=INKEY$:IFB=""THEN12010ELSERETURN
```

****** 32K DISK Sirius Adventure ******

TRS-80/SYSTEM-80

```
100 REM:        Sirius Adventure
110 REM:        Written on the SIRIUS, April 1983
120 REM:        adapted for TRS-80 L2 16K MODEL I  29/6/83
130 REM:        (C) May 1983  Mladen Bauk.
140 REM:
150 REM:        10 Burt st.
160 REM:        Kalamunda
170 REM:        W.A.
180 REM:        6076            Phone: (09) 293 2709
190 REM:
200 CLEAR 200: DEFINT A-Z: VB=22: ND=26: L=21: OB=6: LN=664
210 CLS: PRINT@24,"Sirius Adventure": DEFSTR P: PM=CHR$(93): PF=
" "
220 PRINT@275,"Press:  <I> nstructions  or"
230 PRINT@347,"<B> egin.": CL$=CHR$(30)
240 DIM A$(VB), B$(ND), L$(L), B(OB): GOSUB 1760
270 A$=INKEY$: IF A$="" THEN 270
280 IF A$="I" THEN 2060
290 IF A$<>"B" THEN 270
300 CLS
310 IF LO=OL THEN 410
320 OL=LO: CLS: PRINT@24,"Sirius Adventure"
330 IF LO>4 AND B(1)<>-1 THEN PRINT: PRINT"     It's too dark to
 see!": GOTO 410
340 PRINT: PRINT"     I am "+L$(LO)
345 PRINT@512,PM+STRING$(62,"-")+CHR$(94);
350 TR=0: PRINT@448,CL$;: PRINT@448,"Visible objects >>> ";
360 FOR I=1 TO OB
370 IF B(I)=LO THEN PRINTB$(I);".  ";: TR=-1
380 NEXT I
400 IF TR<>-1 THEN PRINT@468,"None.";
410 PRINT@640,"----"+CHR$(94)+" What should I do?";CL$;: C$="":G
OSUB 2300: PRINT: PRINT
415 IF C$="" THEN PRINT"Huh?": GOTO 410
420 FOR I=1 TO LEN(C$): IF ASC( MID$( C$, I, 1))=32 THEN 440
430 NEXT I: GOTO 450
440 LE$=LEFT$( C$, I-1): RI$=MID$( C$, I+1, LEN(C$)-LEN(LE$)-1):
GOTO 460
450 LE$=LEFT$(C$, I): RI$=""
460 L=LEN(LE$): IF RI$="" THEN R=-1 ELSE R=LEN(RI$)
470 FOR I=1 TO VB: IF L>LEN(A$(I)) THEN 490
480 IF LE$<>LEFT$(A$(I),L) THEN 490 ELSE 510
490 NEXT I
500 IF C$<>"" THEN PRINT"I don't understand "CHR$(34);C$;CHR$(34
)", check my vocabulary.": GOTO 410
510 IF R=-1 THEN 560
520 FOR J=1 TO ND
530 IF RI$<>B$(J) THEN NEXT J ELSE 560
540 PRINT"I don't understand "CHR$(34);RI$;CHR$(34)", check my v
ocabulary.": GOTO 410
560 ON I GOSUB 590,590,590,590,1060,1110,1110,1110,1190,1190,119
0,1190,1230,1320,1230,1530,1230,1410,1460,1600,1680,2400
570 IF I>4 AND I<13 THEN 345
580 IF I=22 THEN 320 ELSE 310
590 IF J<OB+1 THEN PRINT"I can't "CHR$(34);A$(I)+" ";RI$;CHR$(34
)"!": GOTO 410
600 J=J-OB: ON J GOTO 730,820,900,950,730,820,900,950,610,640,67
0,700,610,640,670,700,1000,1030,1000,1030
610 IF LO=13 THEN LO=11 ELSE GOSUB 2260
620 RETURN
640 IF LO=12 THEN LO=11 ELSE IF LO=14 THEN LO=15 ELSE GOSUB 2260
650 RETURN
670 IF LO=11 THEN LO=12 ELSE IF LO=15 THEN LO=14 ELSE GOSUB 2260
680 RETURN
700 IF LO=11 THEN LO=13 ELSE GOSUB 2260
710 RETURN
730 IF LO=2 THEN LO=1 ELSE IF LO=5 THEN LO=4 ELSE IF LO=6 THEN L
O=5
740 IF LO=7 THEN LO=9 ELSE IF LO=11 THEN LO=7
750 IF LO=16 AND B(4)=-1 THEN GOSUB 2270
760 IF LO=16 AND B(4)<>-1 THEN LO=17
770 IF LO=18 AND B(5)=-1 THEN LO=19
780 IF LO=18 AND B(5)<>-1 THEN GOSUB 2270
785 IF LO=15 THEN LO=16
790 IF LO=OL THEN GOSUB 2260
800 RETURN
820 IF LO=1 THEN LO=2 ELSE IF LO=4 THEN LO=5 ELSE IF LO=5 THEN L
O=6
830 IF LO=9 THEN LO=7 ELSE IF LO=7 THEN LO=11 ELSE IF LO=16 THEN
 LO=15
840 IF LO=17 THEN LO=16
850 IF LO=19 AND B(5)=-1 THEN LO=18
860 IF LO=19 AND B(5)<>-1 THEN GOSUB 2270
870 IF LO=OL THEN GOSUB 2260
880 RETURN
900 IF LO=3 THEN LO=2 ELSE IF LO=4 THEN LO=3 ELSE IF LO=10 THEN
LO=7
910 IF LO=7 THEN LO=8 ELSE IF LO=19 THEN LO=20 ELSE IF LO=20 THE
```

```
N LO=21
920 IF LO=OL THEN GOSUB 2260
930 RETURN
950 IF LO=2 THEN LO=3 ELSE IF LO=3 THEN LO=4 ELSE IF LO=7 THEN L
O=10
960 IF LO=8 THEN LO=7 ELSE IF LO=20 THEN LO=19 ELSE IF LO=21 THE
N LO=20
970 IF LO=OL THEN GOSUB 2260
980 RETURN
1000 IF LO=7 THEN LO=6 ELSE IF LO=18 THEN LO=17 ELSE GOSUB 2260
1010 RETURN
1030 IF LO=6 THEN LO=7 ELSE IF LO=17 THEN LO=18 ELSE GOSUB 2260
1040 RETURN
1060 IF J=0 THEN J=3
1070 IF J<>2 THEN PRINT"I can't eat that, stupid.": RETURN
1075 IF J=2 AND B(J)=0 THEN PRINT"I already ate it.":RETURN
1080 IF J=2 THEN PRINT"Munch, chomp, <BURP> -- the cream bun was
 delicious!": B(2)=0: RETURN
1090 PRINT"ERROR": STOP
1110 IF J>OB THEN PRINT"I can't "CHR$(34);C$;CHR$(34)".": RETURN
1115 IF B(J)=-1 THEN PRINT"I already have it!": RETURN
1120 IF B(J)<>LO THEN PRINT"I can't see the "B$(J)" here.":RETUR
N
1130 IT=1: FOR I9=1 TO OB: IF B(I9)=-1 THEN IT=IT+1: NEXT I9 ELS
E NEXT I9
1140 IF IT>3 THEN PRINT"I am carrying too much, check inventory.
":RETURN
1150 PRINT"Ok.  I add a "B$(J)" to my inventory."
1160 B(J)=-1: RETURN
1190 IF J>OB THEN PRINT"I can't "CHR$(34);C$;CHR$(34)".": RETURN
1200 IF B(J)<>-1 THEN PRINT"I don't have a "RI$: RETURN
1210 B(J)=LO: PRINT"Ok": RETURN
1230 IF J>OB THEN PRINT"I don't see anything special.": RETURN
1240 IF B(J)<>-1 THEN PRINT"I am not carrying a "B$(J): RETURN
1250 ON J GOTO 1260,1270,1280,1280,1280,1290
1260 PRINT"It burns brightly.": RETURN
1270 PRINT"It looks tasty!": RETURN
1280 PRINT"Magic seems to emanate from the "B$(J): RETURN
1290 PRINT"Its beautiful!": RETURN
1320 IF J>OB THEN PRINT"You are being silly.": RETURN
1330 IF B(J)<>-1 THEN PRINT"I don't have the "B$(J)".": RETURN
1340 IF J<>3 THEN PRINT"Waving the "B$(J)" is not very rewarding
.": RETURN
1350 PRINT"The room dims and blurs, and...";
1360 FOR I=1 TO 1000: NEXT I
1370 IF LO=13 THEN LO=14 ELSE IF LO=14 THEN LO=13 ELSE PRINT"not
hing happens.": RETURN
1380 PRINT"I am magically transported!": FOR I=1 TO 1000: NEXT I
: RETURN
1410 PRINT"Confirm  <Y/N> ?";: C$="": PRINT@LN,CL$: GOSUB 2300
1420 IF C$="Y" THEN CLS: END
1430 IF C$<>"N" THEN 1410
1440 PRINT:PRINT:PRINT"Confirm  <CANCELLED>": RETURN
1460 IN=0: FOR I9=4 TO 6
1470 IF B(I9)=1 THEN IN=IN+20
1480 NEXT I9
1490 IF IN=60 THEN PRINT"Fantastic! you have solved the adventur
e!"
1500 PRINT"You have"IN"points out of a possible 60."
1510 IF IN=60 THEN END
1520 RETURN
1530 PRINT"I am carrying >>> ";
1540 IN=0: FOR I9=1 TO OB
1550 IF B(I9)=-1 THEN PRINT"A "B$(I9);".  ";: IN=-1
1560 NEXT I9
1570 IF IN<>-1 THEN PRINT"Nothing at all.": RETURN
1580 RETURN
1600 PRINT"Ready disk...press <ENTER>"
1610 IF PEEK(15359)<>1 THEN 1610
1620 CLOSE #1
1630 OPEN "O",#1,"URLORD"
1640 FOR I9=1 TO OB: PRINT#1,B(I);: NEXT I9
1650 PRINT#1,LO
1660 RETURN
1680 PRINT"Ready disk...press <ENTER>"
1690 IF PEEK(15359)<>1 THEN 1690
1700 CLOSE #1
1710 OPEN "I",#1,"URLORD"
1720 FOR I9=1 TO OB: INPUT#1,B(I): NEXT I9
1730 INPUT#1,LO
1740 RETURN
1760 LO=1
1770 FOR I=1 TO VB: READ A$(I): NEXT I
1780 FOR I=1 TO ND: READ B$(I): NEXT I
1790 DATA GO,WALK,RUN,CRAWL,EAT,GET,TAKE,GRAB,DROP,THROW,PUT,LEA
VE,LOOK,WAVE,EXAMINE,INVENTORY,INSPECT,QUIT,SCORE,SAVE,LOAD,VOCA
BULARY
1800 DATA LAMP,BUN,ROD,RING,STATUE,CROWN,N,S,W,E,NORTH,SOUTH,WES
T,EAST,NW,NE,SW,SE,NORTHWEST,NORTHEAST,SOUTHWEST,SOUTHEAST,UP,DO
WN,U,D
1810 DATA 1,6,9,8,12,21
1820 FOR I=1 TO OB: READ B(I): NEXT I
1830 FOR I=1 TO L: READ L$(I): NEXT I: RETURN
1840 DATA "at a plateau near a cliff.  A rocky
path leads south.
Some obvious exits: South."
1850 DATA "on a rocky path leading north and
curving to the east.
Some obvious exits: North. East."
1860 DATA "at the entrance to a dark cave.
A rocky path to the west curves north.  There
is a slight breeze.
Some obvious exits: West. East."
1870 DATA "just inside a dark cave.  Light
comes from an entrance to the west.  There is
a dank, mouldy smell.  A tunnel leads south.
Some obvious exits: West. South."
1880 DATA "in a low north/south tunnel.
Some obvious exits: North. South."
1890 DATA "in an oval cavern.  There is a
forbidding stone staircase here.
Some obvious exits: North. Down."
1900 DATA "in a high, square cave with walls
of frozen ice.  There are passages in many directions.
Some obvious exits: North. South. West. East. Up."
1910 DATA "in a triangular side-chamber.
Some obvious exits: East."
1920 DATA "in a musty-smelling alcove.
Some obvious exits: South."
1930 DATA "in an eerie chamber - small
```

```
squealing sounds come from the walls.
Some obvious exits: West."
1940 DATA "in an enormous cave.  There is
a double pillar of green stone down the centre.
Some obvious exits: North. Southwest. Southeast."
1950 DATA "in a malodourous tunnel.
Some obvious exits: Northeast."
1960 DATA "in a room in which the only VISIBLE
exit is the way I came in.
Some obvious exits: Northwest."
1970 DATA "in a secret room, reached only by
magical means.
Some obvious exits: Northeast."
1980 DATA "in a octagonal room.
Some obvious exits: North. Southwest."
1990 DATA "in an enormous misty cavern.  Mist
obscures the ceiling.
Some obvious exits: North. South."
2000 DATA "in a tiny box-shaped room.
Door leads south and stairs lead down.
Some obvious exits: South. Down."
2010 DATA "in a strange room.  there
is a faint whiff of chlorine.
Some obvious exits: North. Up."
2020 DATA "in a steamy chamber, with
warm walls.
Some obvious exits: West. South."
2030 DATA "in a large room, littered
with alabaster slabs.
Some obvious exits: West. East."
2040 DATA "in the throne room of the
evil Urlord!  A low door leads east.
Some obvious exits: East."
2060 CLS:PRINT:PRINT"     Your quest is to explore the cave of t
he evil Urlord, and"
2070 PRINT"bring back to the edge of the cliff the following val
uables:"
2080 PRINT"          1. The white gold ring."
2090 PRINT"          2. The sacred silver statue."
2100 PRINT"          3. The jewelled crown of the Urlord."
2110 PRINT:PRINT
2120 PRINT"Be careful...":PRINT:PRINT:PRINT
2130 PRINT"                  Press  <C> ontinue."
2140 FOR I=1 TO 4000
2150 A$=INKEY$: IF A$="" THEN 2240
2160 IF A$<>"C" THEN 2150
2170 GOTO 310
2240 NEXT I
2250 GOTO 310
2260 PRINT"You cannot go in that direction.": RETURN
2270 PRINT"An invisible force prevents you from passing."
2280 FOR I=1 TO 1000: NEXT I
2290 RETURN
2300 PRINT@LN+LEN(C$),PM;
2310 A$=INKEY$: IF A$="" THEN 2310
2320 PRINT@LN+LEN(C$),PF;: A=ASC(A$)
2325 IF A>31 THEN 2380
2330 IF A=8 AND LEN(C$)>0 THEN C$=LEFT$(C$,LEN(C$)-1): PRINT@LN,
CL$;:PRINT@LN,C$;: GOTO 2300 ELSE IF A=8 THEN 2300
2340 IF A=13 THEN X=FRE(A$): RETURN
2350 IF A=10 THEN A$=CHR$(92) ELSE IF A=27 THEN A$="@"
2360 IF A=9 THEN A$=CHR$(187) ELSE IF A=31 THEN A$="%"
2370 IF A=24 THEN C$="": PRINT@LN,CL$;: GOTO 2300
2380 C$=C$+A$: IF LEN(C$)>20 THEN RETURN
2390 PRINT@LN,C$;: GOTO 2300
2400 CLS:PRINT@22,A$(22):PRINT@192,;
2410 FOR I9=1 TO VB: PRINT A$(I9),: NEXT I9
2420 A$=INKEY$:IF A$="" THEN 2420 ELSE RETURN
```

****** LII/16K Sharemarket ******

TRS-80/SYSTEM-80

```
10 REM SHAREMARKET-R.BURLING-14/6/81
20 CLS:PRINT@460,CHR$(23)"****SHAREMARKET****":FORN=1TO200:GOSUB
260:NEXTN:PRINTCHR$(28):CLS
30 PRINT:PRINT"DO YOU REQUIRE INSTRUCTIONS (Y OR N)"
40 ZZ$=INKEY$:IFZZ$=""THEN40ELSEIFZZ$="Y"THEN50ELSE100
50 CLS:PRINT"THIS GAME IS FOR ONE TO FOUR PLAYERS. EACH INVESTOR
 PITS THEIR":PRINT:PRINT"SKILL AGAINST THE MARKETS (YOUR COMPUTE
R). YOU ARE ABLE TO":PRINT:PRINT"PURCHASE OR SELL SELECT SHARES
(OR PAY PENALTIES) AFTER EACH":PRINT
60 PRINT"TURN. IN EACH CASE YOU WILL ONLY BE ABLE TO DEAL IN THE
":PRINT:PRINT"COMPANY LISTED, OR TO PAY THE COSTS GIVEN.":PRINT:
PRINT"EACH INVESTOR STARTS WITH $2000.":PRINT
70 PRINT"VALUE OF EACH COMPANY IS RANDOMLY SET WITHIN GIVEN PARA
METERS.":PRINT:GOSUB2290:CLS:PRINT"****  PLEASE NOTE  ****":PRIN
T:PRINT"WHEN INVESTOR CONTROLLED TRANSACTIONS TAKE PLACE, THE NE
XT":PRINT
80 PRINT"STEP WILL FOLLOW THE PRESSING OF 'ENTER'.":PRINT:PRINT"
WHEN A MARKET CONTROLLED TRANSACTION TAKES PLACE THERE WILL":PRI
NT:PRINT"BE A TIME DELAY BEFORE AN AUTOMATIC ADVANCEMENT."
90 PRINT:PRINT"MANY INPUTS WILL NOT REQUIRE YOU TO PRESS 'ENTER'
BUT WILL":PRINT:PRINT"MOVE ON IMMEDIATELY YOU INPUT THE NUMBER S
ELECTED.":PRINT:PRINT"HAVE FUN AND THE BEST OF LUCK.":GOSUB2290
100 CLS:AY=2000:BY=2000:CY=2000:DY=2000:AZ=1:BZ=1:CZ=1:DZ=1
110 A=130:A$=" 1. GOFAR PETROLEUM ":B=60:B$=" 2. EASYWEAR SHOES
"
120 C=45:C$=" 3. BUTCHER PIES     ":D=30:D$=" 4. TEARES PRESS    "
130 E=30:E$=" 5. TICTOC CLOCKS    ":F=45:F$=" 6. FOODTOWN        "
140 G=60:G$=" 7. BANK OF TRS      ":H=130:H$=" 8. GEM MINERALS   "
150 I$="PAY STOCKBROKER FEE OF $100":J$="PAY STOCKBROKER FEE OF
$10 PER SHARE - WHICH IS $":K$="PAY ANNUAL SUBSCRIPTION TO MARKE
T BULLETIN OF$25"
160 L$="SHARES ARE AVAILABLE IN ":O$="YOUR ACCOUNT BALANCE IS $"
:R$="DIVIDENDS ARE NOW PAYED. THE VALUE IS $":T$="WHICH COMPANY
WILL YOU SELL  FROM "
170 M$="THIS IS THE CURRENT MARKET VALUE ":N$="AND SHARES HELD B
Y ":Q$="YOUR ASSETS ARE $":P$=" BONUS SHARES GAINED IN ":S$="TOT
AL NO. OF SHARES HELD: "
180 CLS:PRINT@450,"HOW MANY INVESTORS ARE INVOLVED (MAX. 4)";
190 GOSUB380:W=VAL(ZZ$):IFW>4GOTO180ELSEGOSUB2310
200 CLS:ONWGOTO210,220,230,240
210 FORQZ=1TO20:GOSUB440:IFAZ=0THEN2430ELSENEXTQZ:GOSUB2400:GOSU
B250:GOTO210
220 FORQZ=1TO10:GOSUB440:GOSUB890:NEXTQZ:GOSUB2400:GOSUB250:GOTO
220
230 FORQZ=1TO4:GOSUB440:GOSUB890:GOSUB1340:NEXTQZ:GOSUB2400:GOSU
```

```
B250:GOTO230
240 FORQZ=1TO4:GOSUB440:GOSUB890:GOSUB1340:GOSUB1790:NEXTQZ:GOSU
B2400:GOSUB250:GOTO240
250 CLS:PRINT@450,CHR$(23)"ONE MOMENT PLEASE.":PRINT@515,CHR$(23
)"SHAREMARKET IS BEING STUDIED.":FORN=1TO100:GOSUB260:NEXTN:RETU
RN
260 Q=RND(32767):S=RND(32767):T=RND(32767):U=RND(32767):RETURN
270 TT=1:CLS:PRINT"WE WISH TO ADVISE YOU THAT YOU HAVE OVERDRAWN
 YOUR ACCOUNT.":PRINT
280 PRINT"YOU MAY SELL SHARES TO GAIN FUNDS (1) OR LIQUIDATE (2)
.":PRINT:PRINT"WHAT IS YOUR CHOICE?"
290 GOSUB380:Z=VAL(ZZ$):ONZGOTO300,360
300 CLS:PRINT"CHOOSE THE COMPANY YOU WISH TO SELL FROM":PRINT:GO
SUB2240
310 ONYGOTO320,330,340,350
320 PRINTO$AY:GOSUB400:PRINTQ$AV:PRINTT$:GOSUB390:ONMGOTO500,530
,560,590,620,650,680,710
330 PRINTO$BY:GOSUB410:PRINTQ$BV:PRINTT$:GOSUB390:ONMGOTO950,980
,1010,1040,1070,1100,1130,1160
340 PRINTO$CY:GOSUB420:PRINTQ$CV:PRINTT$:GOSUB390:ONMGOTO1400,14
30,1460,1490,1520,1550,1580,1610
350 PRINTO$DY:GOSUB430:PRINTQ$DV:PRINTT$:GOSUB390:ONMGOTO1850,18
80,1910,1940,1970,2000,2030,2060
360 FORP=1TO10:CLS:PRINT@450,CHR$(23)"YOU ARE LIQUIDATED!!!!!!":
FORN=1TO250:NEXTN:PRINTCHR$(28):CLS:FORN=1TO125:NEXTN:NEXTP
370 DZ=0:RETURN
380 ZZ$=INKEY$:IFZZ$=""THEN380ELSERETURN
390 GOSUB380:M=VAL(ZZ$):RETURN
400 AV=AY+AA*A+AB*B+AC*C+AD*D+AE*E+AF*F+AG*G+AH*H:RETURN
410 BV=BY+BA*A+BB*B+BC*C+BD*D+BE*E+BF*F+BG*G+BH*H:RETURN
420 CV=CY+CA*A+CB*B+CC*C+CD*D+CE*E+CF*F+CG*G+CH*H:RETURN
430 DV=DY+DA*A+DB*B+DC*C+DD*D+DE*E+DF*F+DG*G+DH*H:RETURN
440 IFAZ=0THEN880
450 CLS:PRINTM$N$AA$:GOSUB2590:GOSUB2250
460 PRINT:PRINTO$AY:GOSUB2890:PRINTS$SA:GOSUB400:PRINTQ$AV:GOSUB
470:RETURN
470 Q=RND(23):ONQGOTO480,480,510,510,540,540,570,570,600,600,630
,630,660,660,690,690,720,730,740,750,750,840,840
480 PRINTL$A$:GOSUB2870:TT=1:ONKGOTO490,500,850
490 GOSUB2880:X=AY-R*A:IFX<0THEN490ELSEAY=X:AA=AA+R:GOTO850
500 GOSUB2880:L=AA-R:IFL<0THEN500ELSEAA=L:AY=AY+R*A:GOTO850
510 PRINTL$B$:GOSUB2870:TT=1:ONKGOTO520,530,850
520 GOSUB2880:X=AY-R*B:IFX<0THEN520ELSEAY=X:AB=AB+R:GOTO850
530 GOSUB2880:L=AB-R:IFL<0THEN530ELSEAB=L:AY=AY+R*B:GOTO850
540 PRINTL$C$:GOSUB2870:TT=1:ONKGOTO550,560,850
550 GOSUB2880:X=AY-R*C:IFX<0THEN550ELSEAY=X:AC=AC+R:GOTO850
560 GOSUB2880:L=AC-R:IFL<0THEN560ELSEAC=L:AY=AY+R*C:GOTO850
570 PRINTL$D$:GOSUB2870:TT=1:ONKGOTO580,590,850
580 GOSUB2880:X=AY-R*D:IFX<0THEN580ELSEAY=X:AD=AD+R:GOTO850
590 GOSUB2880:L=AD-R:IFL<0THEN590ELSEAD=L:AY=AY+R*D:GOTO850
600 PRINTL$E$:GOSUB2870:TT=1:ONKGOTO610,620,850
610 GOSUB2880:X=AY-R*E:IFX<0THEN610ELSEAY=X:AE=AE+R:GOTO850
620 GOSUB2880:L=AE-R:IFL<0THEN620ELSEAE=L:AY=AY+R*E:GOTO850
630 PRINTL$F$:GOSUB2870:TT=1:ONKGOTO640,650,850
640 GOSUB2880:X=AY-R*F:IFX<0THEN640ELSEAY=X:AF=AF+R:GOTO850
650 GOSUB2880:L=AF-R:IFL<0THEN650ELSEAF=L:AY=AY+R*F:GOTO850
660 PRINTL$G$:GOSUB2870:TT=1:ONKGOTO670,680,850
670 GOSUB2880:X=AY-R*G:IFX<0THEN670ELSEAY=X:AG=AG+R:GOTO850
680 GOSUB2880:L=AG-R:IFL<0THEN680ELSEAG=L:AY=AY+R*G:GOTO850
690 PRINTL$H$:GOSUB2870:TT=1:ONKGOTO700,710,850
700 GOSUB2880:X=AY-R*H:IFX<0THEN700ELSEAY=X:AH=AH+R:GOTO850
710 GOSUB2880:L=AH-R:IFL<0THEN710ELSEAH=L:AY=AY+R*H:GOTO850
720 PRINTI$:AY=AY-100:TT=10:GOTO850
730 S1=SA*10:PRINTJ$S1:AY=AY-S1:TT=10:GOTO850
740 PRINTK$:AY=AY-25:TT=10:GOTO850
750 T=RND(8):U=RND(3):TT=10:ONTGOTO760,770,780,790,800,810,820,8
30
760 V=AA*U:AA=AA+V:PRINTV;P$A$:GOTO850
770 V=AB*U:AB=AB+V:PRINTV;P$B$:GOTO850
780 V=AC*U:AC=AC+V:PRINTV;P$C$:GOTO850
790 V=AD*U:AD=AD+V:PRINTV;P$D$:GOTO850
800 V=AE*U:AE=AE+V:PRINTV;P$E$:GOTO850
810 V=AF*U:AF=AF+V:PRINTV;P$F$:GOTO850
820 V=AG*U:AG=AG+V:PRINTV;P$G$:GOTO850
830 V=AH*U:AH=AH+V:PRINTV;P$H$:GOTO850
840 QA=RND(3):TT=10:QB=(AA+AB+AC+AD+AE+AF+AG+AH)*QA:PRINTR$QB:AY
=AY+QB
850 GOSUB2300:IFAY>=0THEN880
860 FORN=1TO500:NEXTN:Y=1
870 GOSUB270:IFZ=2AZ=0:IFZ=2AY=0:IFZ=2GOTO880ELSEONMGOTO500,530,
560,590,620,650,680,710
880 Z=0:RETURN
890 IFBZ=0THEN1330
900 CLS:PRINTM$N$BB$:GOSUB2590:GOSUB2260
910 PRINT:PRINTO$BY:GOSUB2900:PRINTS$SB:GOSUB410:PRINTQ$BV:GOSUB
920:RETURN
920 Q=RND(23):ONQGOTO930,930,960,960,990,990,1020,1020,1050,1050
,1080,1080,1110,1110,1140,1140,1170,1180,1190,1200,1200,1290,129
0
930 PRINTL$A$:GOSUB2870:TT=1:ONKGOTO940,950,1300
940 GOSUB2880:X=BY-R*A:IFX<0THEN940ELSEBY=X:BA=BA+R:GOTO1300
950 GOSUB2880:L=BA-R:IFL<0THEN950ELSEBA=L:BY=BY+R*A:GOTO1300
960 PRINTL$B$:GOSUB2870:TT=1:ONKGOTO970,980,1300
970 GOSUB2880:X=BY-R*B:IFX<0THEN970ELSEBY=X:BB=BB+R:GOTO1300
980 GOSUB2880:L=BB-R:IFL<0THEN980ELSEBB=L:BY=BY+R*B:GOTO1300
990 PRINTL$C$:GOSUB2870:TT=1:ONKGOTO1000,1010,1300
1000 GOSUB2880:X=BY-R*C:IFX<0THEN1000ELSEBY=X:BC=BC+R:GOTO1300
1010 GOSUB2880:L=BC-R:IFL<0THEN1010ELSEBC=L:BY=BY+R*C:GOTO1300
1020 PRINTL$D$:GOSUB2870:TT=1:ONKGOTO1030,1040,1300
1030 GOSUB2880:X=BY-R*D:IFX<0THEN1030ELSEBY=X:BD=BD+R:GOTO1300
1040 GOSUB2880:L=BD-R:IFL<0THEN1040ELSEBD=L:BY=BY+R*D:GOTO1300
1050 PRINTL$E$:GOSUB2870:TT=1:ONKGOTO1060,1070,1300
1060 GOSUB2880:X=BY-R*E:IFX<0THEN1060ELSEBY=X:BE=BE+R:GOTO1300
1070 GOSUB2880:L=BE-R:IFL<0THEN1070ELSEBE=L:BY=BY+R*E:GOTO1300
1080 PRINTL$F$:GOSUB2870:TT=1:ONKGOTO1090,1100,1300
1090 GOSUB2880:X=BY-R*F:IFX<0THEN1090ELSEBY=X:BF=BF+R:GOTO1300
1100 GOSUB2880:L=BF-R:IFL<0THEN1100ELSEBF=L:BY=BY+R*F:GOTO1300
1110 PRINTL$G$:GOSUB2870:TT=1:ONKGOTO1120,1130,1300
1120 GOSUB2880:X=BY-R*G:IFX<0THEN1120ELSEBY=X:BG=BG+R:GOTO1300
1130 GOSUB2880:L=BG-R:IFL<0THEN1130ELSEBG=L:BY=BY+R*G:GOTO1300
1140 PRINTL$H$:GOSUB2870:TT=1:ONKGOTO1150,1160,1300
1150 GOSUB2880:X=BY-R*H:IFX<0THEN1150ELSEBY=X:BH=BH+R:GOTO1300
1160 GOSUB2880:L=BH-R:IFL<0THEN1160ELSEBH=L:BY=BY+R*H:GOTO1300
1170 PRINTI$:BY=BY-100:TT=10:GOTO1300
1180 S2=SB*10:PRINTJ$S2:BY=BY-S2:TT=10:GOTO1300
1190 PRINTK$:BY=BY-25:TT=10:GOTO1300
1200 T=RND(8):U=RND(3):TT=10:ONTGOTO1210,1220,1230,1240,1250,126
0,1270,1280
1210 V=BA*U:BA=BA+V:PRINTV;P$A$:GOTO1300
1220 V=BB*U:BB=BB+V:PRINTV;P$B$:GOTO1300
```

```
1230 V=BC*U:BC=BC+V:PRINTV;P$C$:GOTO1300
1240 V=BD*U:BD=BD+V:PRINTV;P$D$:GOTO1300
1250 V=BE*U:BE=BE+V:PRINTV;P$E$:GOTO1300
1260 V=BF*U:BF=BF+V:PRINTV;P$F$:GOTO1300
1270 V=BG*U:BG=BG+V:PRINTV;P$G$:GOTO1300
1280 V=BH*U:BH=BH+V:PRINTV;P$H$:GOTO1300
1290 QA=RND(3):TT=10:QB=(BA+BB+BC+BD+BE+BF+BG+BH)*QA:PRINTR$QB:B
Y=BY+QB
1300 GOSUB2300:IFBY>=0THEN880
1310 FORN=1TO500:NEXTN:Y=2
1320 GOSUB270:IFZ=2BZ=0:IFZ=2BY=0:IFZ=2GOTO1330ELSEONMGOTO950,98
0,1010,1040,1070,1100,1130,1160
1330 Z=0:RETURN
1340 IFCZ=0THEN1780
1350 CLS:PRINTM$N$CC$:GOSUB2590:GOSUB2270
1360 PRINT:PRINTO$CY:GOSUB2910:PRINTS$SC:GOSUB420:PRINTQ$CV:GOSU
B1370:RETURN
1370 Q=RND(23):ONQGOTO1380,1380,1410,1410,1440,1440,1470,1470,15
00,1500,1530,1530,1560,1560,1590,1590,1620,1630,1640,1650,1650,1
740,1740
1380 PRINTL$A$:GOSUB2870:TT=1:ONKGOTO1390,1400,1750
1390 GOSUB2880:X=CY-R*A:IFX<0THEN1390ELSECY=X:CA=CA+R:GOTO1750
1400 GOSUB2880:L=CA-R:IFL<0THEN1400ELSECA=L:CY=CY+R*A:GOTO1750
1410 PRINTL$B$:GOSUB2870:TT=1:ONKGOTO1420,1430,1750
1420 GOSUB2880:X=CY-R*B:IFX<0THEN1420ELSECY=X:CB=CB+R:GOTO1750
1430 GOSUB2880:L=CB-R:IFL<0THEN1430ELSECA=L:CY=CY+R*B:GOTO1750
1440 PRINTL$C$:GOSUB2870:TT=1:ONKGOTO1450,1460,1750
1450 GOSUB2880:X=CY-R*C:IFX<0THEN1450ELSECY=X:CC=CC+R:GOTO1750
1460 GOSUB2880:L=CC-R:IFL<0THEN1460ELSECC=L:CY=CY+R*C:GOTO1750
1470 PRINTL$D$:GOSUB2870:TT=1:ONKGOTO1480,1490,1750
1480 GOSUB2880:X=CY-R*D:IFX<0THEN1480ELSECY=X:CD=CD+R:GOTO1750
1490 GOSUB2880:L=CD-R:IFL<0THEN1490ELSECD=L:CY=CY+R*D:GOTO1750
1500 PRINTL$E$:GOSUB2870:TT=1:ONKGOTO1510,1520,1750
1510 GOSUB2880:X=CY-R*E:IFX<0THEN1510ELSECY=X:CE=CE+R:GOTO1750
1520 GOSUB2880:L=CE-R:IFL<0THEN1520ELSECE=L:CY=CY+R*E:GOTO1750
1530 PRINTL$F$:GOSUB2870:TT=1:ONKGOTO1540,1550,1750
1540 GOSUB2880:X=CY-R*F:IFX<0THEN1540ELSECY=X:CF=CF+R:GOTO1750
1550 GOSUB2880:L=CF-R:IFL<0THEN1550ELSECF=L:CY=CY+R*F:GOTO1750
1560 PRINTL$G$:GOSUB2870:TT=1:ONKGOTO1570,1580,1750
1570 GOSUB2880:X=CY-R*G:IFX<0THEN1570ELSECY=X:CG=CG+R:GOTO1750
1580 GOSUB2880:L=CG-R:IFL<0THEN1580ELSECG=L:CY=CY+R*G:GOTO1750
1590 PRINTL$H$:GOSUB2870:TT=1:ONKGOTO1600,1610,1750
1600 GOSUB2880:X=CY-R*H:IFX<0THEN1600ELSECY=X:CH=CH+R:GOTO1750
1610 GOSUB2880:L=CH-R:IFL<0THEN1610ELSECH=L:CY=CY+R*H:GOTO1750
1620 PRINTI$:CY=CY-100:TT=10:GOTO1750
1630 S3=SC*10:PRINTJ$S3:CY=CY-S3:TT=10:GOTO1750
1640 PRINTK$:CY=CY-25:TT=10:GOTO1750
1650 T=RND(8):U=RND(3):TT=10:ONTGOTO1660,1670,1680,1690,1700,171
0,1720,1730
1660 V=CA*U:CA=CA+V:PRINTV;P$A$:GOTO1750
1670 V=CB*U:CB=CB+V:PRINTV;P$B$:GOTO1750
1680 V=CC*U:CC=CC+V:PRINTV;P$C$:GOTO1750
1690 V=CD*U:CD=CD+V:PRINTV;P$D$:GOTO1750
1700 V=CE*U:CE=CE+V:PRINTV;P$E$:GOTO1750
1710 V=CF*U:CF=CF+V:PRINTV;P$F$:GOTO1750
1720 V=CG*U:CG=CG+V:PRINTV;P$G$:GOTO1750
1730 V=CH*U:CH=CH+V:PRINTV;P$H$:GOTO1750
1740 QA=RND(3):TT=10:QB=(CA+CB+CC+CD+CE+CF+CG+CH)*QA:PRINTR$QB:C
Y=CY+QB
1750 GOSUB2300:IFCY>=0THEN880
1760 FORN=1TO500:NEXTN:Y=3
1770 GOSUB270:IFZ=2CZ=0:IFZ=2CY=0ELSEONMGOTO1400,1430,1460,1490,
1520,1550,1580,1610
1780 Z=0:RETURN
1790 IFDZ=0THEN1330
1800 CLS:PRINTM$N$DD$:GOSUB2590:GOSUB2280
1810 PRINT:PRINTO$DY:GOSUB2920:PRINTS$SD:GOSUB430:PRINTQ$DV:GOSU
B1820:RETURN
1820 Q=RND(23):ONQGOTO1830,1830,1860,1860,1890,1890,1920,1920,19
50,1950,1980,1980,2010,2010,2040,2040,2070,2080,2090,2100,2100,2
190,2190
1830 PRINTL$A$:GOSUB2870:TT=1:ONKGOTO1840,1850,2200
1840 GOSUB2880:X=DY-R*A:IFX<0THEN1840ELSEDY=X:DA=DA+R:GOTO2200
1850 GOSUB2880:L=DA-R:IFL<0THEN1850ELSEDA=L:DY=DY+R*A:GOTO2200
1860 PRINTL$B$:GOSUB2870:TT=1:ONKGOTO1870,1880,2200
1870 GOSUB2880:X=DY-R*B:IFX<0THEN1870ELSEDY=X:DB=DB+R:GOTO2200
1880 GOSUB2880:L=DB-R:IFL<0THEN1880ELSEDB=L:DY=DY+R*B:GOTO2200
1890 PRINTL$C$:GOSUB2870:TT=1:ONKGOTO1900,1910,2200
1900 GOSUB2880:X=DY-R*C:IFX<0THEN1900ELSEDY=X:DC=DC+R:GOTO2200
1910 GOSUB2880:L=DC-R:IFL<0THEN1910ELSEDC=L:DY=DY+R*C:GOTO2200
1920 PRINTL$D$:GOSUB2870:TT=1:ONKGOTO1930,1940,2200
1930 GOSUB2880:X=DY-R*D:IFX<0THEN1930ELSEDY=X:DD=DD+R:GOTO2200
1940 GOSUB2880:L=DD-R:IFL<0THEN1940ELSEDD=L:DY=DY+R*D:GOTO2200
1950 PRINTL$E$:GOSUB2870:TT=1:ONKGOTO1960,1970,2200
1960 GOSUB2880:X=DY-R*E:IFX<0THEN1960ELSEDY=X:DE=DE+R:GOTO2200
1970 GOSUB2880:L=DE-R:IFL<0THEN1970ELSEDE=L:DY=DY+R*E:GOTO2200
1980 PRINTL$F$:GOSUB2870:TT=1:ONKGOTO1990,2000,2200
1990 GOSUB2880:X=DY-R*F:IFX<0THEN1990ELSEDY=X:DF=DF+R:GOTO2200
2000 GOSUB2880:L=DF-R:IFL<0THEN2000ELSEDF=L:DY=DY+R*F:GOTO2200
2010 PRINTL$G$:GOSUB2870:TT=1:ONKGOTO2020,2030,2200
2020 GOSUB2880:X=DY-R*G:IFX<0THEN2020ELSEDY=X:DG=DG+R:GOTO2200
2030 GOSUB2880:L=DG-R:IFL<0THEN2030ELSEDG=L:DY=DY+R*G:GOTO2200
2040 PRINTL$H$:GOSUB2870:TT=1:ONKGOTO2050,2060,2200
2050 GOSUB2880:X=DY-R*H:IFX<0THEN2050ELSEDY=X:DH=DH+R:GOTO2200
2060 GOSUB2880:L=DH-R:IFL<0THEN2060ELSEDH=L:DY=DY+R*H:GOTO2200
2070 PRINTI$:DY=DY-100:TT=10:GOTO2200
2080 S4=SD*10:PRINTJ$S4:DY=DY-S4:TT=10:GOTO2200
2090 PRINTK$:DY=DY-25:TT=10:GOTO2200
2100 T=RND(8):U=RND(3):TT=10:ONTGOTO2110,2120,2130,2140,2150,216
0,2170,2180
2110 V=DA*U:DA=DA+V:PRINTV;P$A$:GOTO2200
2120 V=DB*U:DB=DB+V:PRINTV;P$B$:GOTO2200
2130 V=DC*U:DC=DC+V:PRINTV;P$C$:GOTO2200
2140 V=DD*U:DD=DD+V:PRINTV;P$D$:GOTO2200
2150 V=DE*U:DE=DE+V:PRINTV;P$E$:GOTO2200
2160 V=DF*U:DF=DF+V:PRINTV;P$F$:GOTO2200
2170 V=DG*U:DG=DG+V:PRINTV;P$G$:GOTO2200
2180 V=DH*U:DH=DH+V:PRINTV;P$H$:GOTO1750
2190 QA=RND(3):TT=10:QB=(DA+DB+DC+DD+DE+DF+DG+DH)*QA:PRINTR$QB:D
Y=DY+QB
2200 GOSUB2300:IFDY>=0THEN2230
2210 FORN=1TO500:NEXTN:Y=4
2220 GOSUB270:IFZ=2DZ=0:IFZ=2DY=0:IFZ=2THEN880ELSEONMGOTO1850,18
80,1910,1940,1970,2000,2030,2060
2230 Z=0:RETURN
2240 ONYGOTO2250,2260,2270,2280
2250 PRINTA$,A,AA:PRINTB$,B,AB:PRINTC$,C,AC:PRINTD$,D,AD:PRINTE$
,E,AE:PRINTF$,F,AF:PRINTG$,G,AG:PRINTH$,H,AH:RETURN
2260 PRINTA$,A,BA:PRINTB$,B,BB:PRINTC$,C,BC:PRINTD$,D,BD:PRINTE$
,E,BE:PRINTF$,F,BF:PRINTG$,G,BG:PRINTH$,H,BH:RETURN
```

```
2270 PRINTA$,A,CA:PRINTB$,B,CB:PRINTC$,C,CC:PRINTD$,D,CD:PRINTE$
,E,CE:PRINTF$,F,CF:PRINTG$,G,CG:PRINTH$,H,CH:RETURN
2280 PRINTA$,A,DA:PRINTB$,B,DB:PRINTC$,C,DC:PRINTD$,D,DD:PRINTE$
,E,DE:PRINTF$,F,DF:PRINTG$,G,DG:PRINTH$,H,DH:RETURN
2290 INPUT"PRESS ENTER TO CONTINUE";I:RETURN
2300 FORN=1TO(TT*200):NEXTN:RETURN
2310 CLS:ONWGOTO2320,2330,2340,2350
2320 GOSUB2360:RETURN
2330 GOSUB2360:GOSUB2370:RETURN
2340 GOSUB2360:GOSUB2370:GOSUB2380:RETURN
2350 GOSUB2360:GOSUB2370:GOSUB2380:GOSUB2390:RETURN
2360 INPUT"PLAYER 1";AA$:RETURN
2370 INPUT"PLAYER 2";BB$:RETURN
2380 INPUT"PLAYER 3";CC$:RETURN
2390 INPUT"PLAYER 4";DD$:RETURN
2400 CLS:PRINT@450,"TO CONTINUE INVESTING ENTER '1'; TO FINISH E
NTER '2'";
2410 GOSUB380:J=VAL(ZZ$):ONJGOTO2420,2430
2420 RETURN
2430 CLS:IFAZ=0THEN2440ELSEAZ=AY+AA*A+AB*B+AC*C+AD*D+AE*E+AF*F+A
G*G+AH*H
2440 IFBZ=0THEN2450ELSEBZ=BY+BA*A+BB*B+BC*C+BD*D+BE*E+BF*F+BG*G+
BH*H
2450 IFCZ=0THEN2460ELSECZ=CY+CA*A+CB*B+CC*C+CD*D+CE*E+CF*F+CG*G+
CH*H
2460 IFDZ=0THEN2470ELSEDZ=DY+DA*A+DB*B+DC*C+DD*D+DE*E+DF*F+DG*G+
DH*H
2470 Z$=" IS WORTH A TOTAL OF $"
2480 ONWGOTO2490,2500,2510,2520
2490 GOSUB2530:GOTO2570
2500 GOSUB2530:GOSUB2540:GOTO2570
2510 GOSUB2530:GOSUB2540:GOSUB2550:GOTO2570
2520 GOSUB2530:GOSUB2540:GOSUB2550:GOSUB2560:GOTO2570
2530 PRINTAA$Z$AZ:PRINT:RETURN
2540 PRINTBB$Z$BZ:PRINT:RETURN
2550 PRINTCC$Z$CZ:PRINT:RETURN
2560 PRINTDD$Z$DZ:PRINT:RETURN
2570 PRINT"THANK YOU FOR PLAYING.":FORN=1TO10000:NEXTN:END
2580 PRINT"THANK YOU FOR PLAYING.":FORN=1TO5000:NEXTN:END
2590 S=RND(10):ONSGOTO2600,2610,2620,2630,2640,2650,2660,2670,26
80,2690
2600 A=A+RND(10):B=B-RND(10):C=C+RND(10):D=D-RND(10):E=E+RND(10)
:F=F+RND(10):G=G+RND(10):H=H-RND(10):GOTO2700
2610 A=A+RND(10):B=B+RND(10):C=C-RND(10):D=D-RND(10):E=E+RND(10)
:F=F-RND(10):G=G-RND(10):H=H-RND(10):GOTO2700
2620 A=A-RND(10):B=B+RND(10):C=C-RND(10):D=D+RND(10):E=E-RND(10)
:F=F-RND(10):G=G-RND(10):H=H-RND(10):GOTO2700
2630 A=A+RND(10):B=B-RND(10):C=C+RND(10):D=D+RND(10):E=E-RND(10)
:F=F+RND(10):G=G-RND(10):H=H+RND(10):GOTO2700
2640 A=A-RND(10):B=B-RND(10):C=C-RND(10):D=D-RND(10):E=E-RND(10)
:F=F-RND(10):G=G-RND(10):H=H-RND(10):GOTO2700
2650 A=A+RND(10):B=B+RND(10):C=C+RND(10):D=D+RND(10):E=E+RND(10)
:F=F+RND(10):G=G+RND(10):H=H+RND(10):GOTO2700
2660 A=A-RND(10):B=B-RND(10):C=C+RND(10):D=D-RND(10):E=E+RND(10)
:F=F+RND(10):G=G-RND(10):H=H+RND(10):GOTO2700
2670 A=A-RND(10):B=B+RND(10):C=C-RND(10):D=D-RND(10):E=E-RND(10)
:F=F-RND(10):G=G+RND(10):H=H+RND(10):GOTO2700
2680 A=A+RND(10):B=B-RND(10):C=C+RND(10):D=D+RND(10):E=E-RND(10)
:F=F+RND(10):G=G-RND(10):H=H+RND(10):GOTO2700
2690 A=A-RND(10):B=B+RND(10):C=C-RND(10):D=D+RND(10):E=E+RND(10)
:F=F-RND(10):G=G+RND(10):H=H-RND(10)
2700 IFA>230THENA=230
2710 IF B>110THENB=110
2720 IFC>75THENC=75
2730 IFD>42THEND=42
2740 IFE>42THENE=42
2750 IFF>75THENF=75
2760 IFG>110THENG=110
2770 IFH>230THENH=230
2780 IFA<30THENA=30
2790 IFB<10THENB=10
2800 IFC<15THENC=15
2810 IFD<18THEND=18
2820 IFE<18THENE=18
2830 IFF<18THENF=18
2840 IFG<10THENG=10
2850 IFH<30THENH=30
2860 RETURN
2870 PRINT"DO YOU WISH TO BUY(1); SELL(2); DO NOTHING(3)?":GOSUB
380:K=VAL(ZZ$):RETURN
2880 INPUT"HOW MANY SHARES";R:RETURN
2890 SA=AA+AB+AC+AD+AE+AF+AG+AH:RETURN
2900 SB=BA+BB+BC+BD+BE+BF+BG+BH:RETURN
2910 SC=CA+CB+CC+CD+CE+CF+CG+CH:RETURN
2920 SD=DA+DB+DC+DD+DE+DF+DG+DH:RETURN
```

****** LII/16K Words And Meanings ******

TRS-80/SYSTEM-80

```
1 '    ******   WORDS & MEANINGS   ******
2 '
3 '      MURRAY J. DIXON
4 '          AQUINAS  COLLEGE, RINGWOOD, VIC.
5 '
6 '
10 CLEAR2000:DIMA$(25,2),P(25),R(25),S(25):DEFINT X,Q,Z
20 CLS:PRINT@340,CHR$(23);"WORD":PRINT@466,"GAMES":GOSUB630
30 CLS:PRINT:PRINT:INPUT"WHAT IS YOUR NAME";NA$
40 PRINT:PRINT"OKAY, ";NA$;" DO YOU WANT INSTRUCTIONS";:INPUT Y1
$
50 IF LEFT$(Y1$,1)="Y" GOSUB650
60 FOR X=1TO25
70 P(X)=0:R(X)=0:S(X)=0:A$(X,1)="":A$(X,2)=""
80 NEXT X
90 PRINT:INPUT"HOW MANY QUESTIONS DO YOU WANT (1 TO 24)";AN:AN=I
NT(AN)
100 IF AN<1 OR AN>24 THEN90
110 CLS
120 IF AN>15 THENPRINT"WAIT A SECOND WHILE I CHOOSE THE WORDS ..
....."
130 ' READ WORDS & MEANINGS INTO A$ ARRAY
140 FOR X=1 TO AN
150 RESTORE
160 Y=RND(24)
170 FOR Z=1 TO X
180 IF P(Z)=Y THEN160
190 NEXT Z
200 P(X)=Y
```

```
210 FOR Z=1 TO Y
220 READ A$,W$
230 NEXT Z
240 READ A$(X,1),A$(X,2)
250 NEXT X
260 ' PRINT WORDS RANDOMLY AT TOP OF SCREEN
270 CLS
280 FOR X=1TO AN
290 Y=RND(AN)
300 FOR Q=1TO X
310 IF R(Q)=Y THEN290
320 NEXT Q
330 R(X)=Y
340 PRINT@16*(X-1)+64,A$(Y,2);
350 NEXT X
360 FOR X=15808 TO 15871:POKE X,140:NEXT X
370 R=0
380 ' CHOOSE QUESTION
390 FOR X=1 TO AN
400 Y=RND(AN)
410 FOR Q=1 TO X
420 IF S(Q)=Y THEN400
430 NEXT Q
440 S(X)=Y
450 PRINT@514,"QUESTION ";X
460 PRINT@645,STRING$(35," ");
470 PRINT@645,A$(Y,1);
480 PRINT@778,STRING$(50," ");
490 PRINT@778,"YOUR ANSWER ------";:INPUT AN$
500 IF AN$=A$(Y,2) THEN540
510 PRINT@900,"SORRY, ";NA$;" THAT'S NOT CORRECT --- TRY AGAIN"
:GOSUB630:GOSUB620
520 IF TR=0 THENR=R+1
530 TR=TR+1:GOTO480
540 PRINT@900,"THAT'S RIGHT, ";NA$;:GOSUB630:GOSUB620
550 TR=0
560 NEXT X
570 CLS:PRINT:PRINT:PRINTNA$;", YOUR SCORE WAS ";AN-R;" OUT OF";
AN
580 PRINT:PRINT:INPUT"WANT TO TRY AGAIN";Y1$
590 IF LEFT$(Y1$,1)="Y" THEN60
600 PRINT:PRINT:PRINT"GOODBYE THEN, ";NA$
610 END
620 PRINT@900,CHR$(31);:RETURN
630 FOR Z=1 TO 1000:NEXT Z:RETURN
640 '    INSTRUCTIONS
650 CLS:PRINT"***** INSTRUCTIONS *****"
660 PRINT:PRINT"I WILL PRINT A LIST OF WORDS AT THE TOP
OF THE SCREEN"
670 PRINT:PRINT"THEN I WILL PRINT A MEANING AND YOU MUST
TYPE IN THE WORD FROM THE LIST THAT MATCHES
THE MEANING"
680 PRINT:PRINT:PRINT:INPUT"'PRESS <NEWLINE> TO BEGIN'";Y1
690 CLS:RETURN
700 DATA A SMALL HORSE,PONY,TO LET DROP,FUMBLE
710 DATA TO CHOKE OR STRANGLE,THROTTLE,TO SING LIKE A BIRD,WARBL
E
720 DATA A ROBBER AT SEA,PIRATE,A LEG  ARM OR WING,LIMB
730 DATA LOGS FASTENED TOGETHER TO FLOAT,RAFT,WANTING VERY MUCH,
EAGER
740 DATA A SMALL CUT OR NICK,NOTCH,A SONG SUNG BY TWO PEOPLE,DUE
T
750 DATA A NUT FROM A TREE,ALMOND,STANDING UP STRAIGHT,UPRIGHT
760 DATA TO BECOME SMALLER,SHRINK,TO GALLOP GENTLY,CANTER
770 DATA TO LOVE VERY MUCH,ADORE,SKIN AND HAIR OF THE HEAD,SCALP
780 DATA TO MAKE INTO LEATHER,TAN,A KIND OF DOG,TERRIER
790 DATA SOMETHING THAT CAN BE BURNED,FUEL,A WREATH OF FLOWERS,G
ARLAND
800 DATA HE DOES TRICKS,CONJURER,TO LOOK AT CLOSELY,EXAMINE
810 DATA WALK WITH SHAKY STEPS,TOTTER,HE REPAIRS PIPES,PLUMBER
820 DATA A PASSING INTEREST IN SOMETHING,CRAZE
```

****** LII/16K Array Utility Demonstration Program ******

```
10 'Basic program to illustrate use of SAVE, LOAD, KILL & NAME
20 'with utility program: ARRAY.
30 DEFINTI,J:DEFDBLD:CLEAR500
40 DIM S(10,10), D(5,5), T$(20)
50 'Assign values to the arrays.
60 FOR I=1 TO 10: FOR J=1 TO 10: S(I,J)=I*J/4: NEXTJ,I
70 FOR I=1 TO 5: FOR J=1 TO 5: D(I,J)=I*J/7#: NEXTJ,I
80 FOR I=1 TO 20: T$(I)= STRING$(10,"T"): NEXTI
90 'Using NAME with a subroutine to print an array
100 N=10: NAME S,G: GOSUB500: NAMEG,S
110 INPUT"Press any key to continue      ";A$
120 'To save an array
130 CLS:PRINT@320,"Saving string array T$:"
140 SAVET$
150 INPUT"Press any key to continue      ";A$
160 CLS:PRINT@320,"Reloading a saved array:"
170 KILLT$     'Erase the original first
180 LOADT2$
190 'Print it to check
200 FOR I=1 TO 20: PRINT T2$(I): NEXTI
210 END
500 FORI=1TON:FORJ=1TON:PRINTG(I,J);:NEXTJ:PRINT:NEXTI:RETURN
```

****** LII/16K ML Array Utility ******

```
7BAB:  20 4F 46 20 41 E5 D5 21 DC 7B 22 A1 41 21 B7 7C
7BBB:  22 89 41 21 1B 7E 22 92 41 21 5A 7E 22 8F 41 AF
7BCB:  32 3E 40 32 3F 40 21 1E 7F CD A7 28 D1 E1 C3 CC
7BDB:  06 F3 C5 D5 DD E5 FD E5 CD 11 7F 11 D7 7F CD A9
7BEB:  7E E5 CD E9 7E E5 DD E1 DD E5 DD 4E 03 DD 46 04
7BFB:  21 05 00 09 22 E0 7F E5 3E 03 DD BE 00 20 16 32
7C0B:  DE 7F DD 7E 05 CB 27 C6 06 32 DF 7F ED 44 4F 06
7C1B:  FF 09 22 E0 7F AF 32 E2 7F CD BC 7E 3A 3E 40 16
7C2B:  00 CD 12 02 CD 87 02 06 04 21 DE 7F 7E CD D9 7E
7C3B:  23 10 F9 C1 E1 3A DE 7F B7 20 1E 7E CD D9 7E 23
7C4B:  0B 78 B1 20 F6 06 0A 7A CD 64 02 10 FB CD F8 01
7C5B:  21 61 7F CD A7 28 C3 13 7E FD 21 00 00 3A DF 7F
7C6B:  47 7E CD D9 7E 23 10 F9 ED 4B E0 7F E5 DD E1 DD
7C7B:  7E 00 CD D9 7E B7 28 17 DD 6E 01 DD 66 02 C5 06
7C8B:  00 4F 09 2B 47 7E CD D9 7E 2B FD 23 10 F7 C1 DD
7C9B:  23 DD 23 DD 23 0B 0B 0B 78 B1 20 D3 FD E5 E1 D5
7CAB:  CD AF 0F 21 75 7F CD A7 28 D1 18 99 F3 C5 D5 DD
7CBB:  E5 FD E5 CD 11 7F 11 D7 7F CD A9 7E E5 3E 01 32
7CCB:  E2 7F CD BC 7E 2A FD 40 E5 AF 32 DE 7F 57 3A 3E
```

```
7CDB:  40 16 00 CD 12 02 CD 96 02 CD E1 7E FE 03 CA AF
7CEB:  7D CD E1 7E CD E1 7E 4F CD E1 7E 47 CD E1 7E 77
7CFB:  23 0B AF B9 CC 2C 02 78 B1 20 F1 CD 35 02 F5 CD
7D0B:  F8 01 F1 BA 28 16 E1 21 8B 7F CD A7 28 3A DE 7F
7D1B:  FE 03 C2 13 7E FD 22 D6 40 C3 13 7E 22 FD 40 DD
7D2B:  E1 3A D8 7F DD 77 01 3A D7 7F DD 77 02 3A 3F 40
7D3B:  FE 00 CA 13 7E 21 CF 7F CD A7 28 DD 4E 00 CB 41
7D4B:  20 4A CB 49 20 4E CB 51 20 52 CB 59 28 B9 21 AC
7D5B:  7F CD A7 28 21 B1 7F CD A7 28 11 05 00 DD 19 DD
7D6B:  E5 E1 46 3E 02 32 AF 40 23 5E 23 56 1B ED 53 21
7D7B:  41 E5 C5 21 D7 7F CD BD 0F CD A7 28 3E 20 CD 3A
7D8B:  03 C1 E1 10 DE 3E 0D CD 3A 03 18 7C 21 9D 7F CD
7D9B:  A7 28 18 C0 21 A2 7F CD A7 28 18 B8 21 A7 7F CD
7DAB:  A7 28 18 B0 32 DE 7F FD 2A D6 40 CD E1 7E 47 CD
7DBB:  E1 7E 6F CD E1 7E 67 E5 2A FD 40 CD E1 7E 77 23
7DCB:  10 F9 C1 CD E1 7E 77 23 FE 00 20 05 77 23 77 18
7DDB:  29 C5 E5 2A A0 40 06 00 4F 09 ED 4B D6 40 B7 ED
7DEB:  42 D2 83 7E 47 2A D6 40 CD E1 7E 77 2B 10 F9 22
7DFB:  D6 40 23 E5 C1 E1 71 23 70 C1 23 0B 0B 0B CD 2C
7E0B:  02 78 B1 20 BE C3 06 7D E1 FD E1 DD E1 D1 C1 C9
7E1B:  C5 D5 DD E5 FD E5 CD 11 7F 11 D7 7F CD A9 7E E5
7E2B:  CD E9 7E E5 23 23 23 4E 23 46 23 09 E5 ED 5B FD
7E3B:  40 CD C7 0B E5 C1 E1 D1 ED 53 FD 40 78 B1 CA 13
7E4B:  7E D5 C5 D9 C1 E1 09 22 FD 40 D9 ED B0 18 B9 C5
7E5B:  D5 DD E5 FD E5 CD 11 7F 11 D7 7F CD A9 7E 11 D9
7E6B:  7F D7 CD A9 7E E5 CD E9 7E 23 3A DA 7F 77 23 3A
7E7B:  D9 7F 77 CD 11 7F 18 90 F1 F1 F1 FD 22 D6 40 CD
7E8B:  F8 01 21 B9 7F CD A7 28 C3 13 7E 21 E3 7F CD A7
7E9B:  28 C3 13 7E F1 21 EF 7F CD A7 28 C3 13 7E 7E 12
7EAB:  13 D7 C8 FE 2C C8 38 01 12 13 D7 C8 FE 2C 20 FA
7EBB:  C9 21 50 7F CD A7 28 CD E3 03 B7 28 FA FE 01 C0
7ECB:  F1 3A E2 7F FE 01 CA 13 7E F1 F1 C3 13 7E 5F 82
7EDB:  57 7B CD 64 02 C9 CD 35 02 5F 82 57 7B C9 B7 2A
7EEB:  FD 40 ED 4B FB 40 ED 42 28 AA C5 E5 C1 E1 3A D7
7EFB:  7F ED B1 28 03 F1 18 93 2B 2B 3A D8 7F BE 28 04
7F0B:  23 23 18 EA 2B C9 E5 AF 06 0B 21 D7 7F 77 23 10
7F1B:  FC E1 C9 1C 1F 2A 20 2A 20 41 52 52 41 59 20 55
7F2B:  54 49 4C 49 54 59 20 20 20 42 59 20 52 20 45 20
7F3B:  54 41 50 4C 49 4E 20 28 43 29 20 31 39 38 31 20
7F4B:  2A 20 2A 0D 00 0D 52 45 41 44 59 20 43 41 53 53
7F5B:  45 54 54 45 0D 00 52 45 43 4F 52 44 49 4E 47 20
7F6B:  43 4F 4D 50 4C 45 54 45 0D 00 20 43 48 41 52 41
7F7B:  43 54 45 52 53 20 52 45 43 4F 52 44 45 44 0D 00
7F8B:  0D 53 41 56 45 2F 4C 4F 41 44 20 45 52 52 4F 52
7F9B:  0D 00 20 53 54 52 00 20 49 4E 54 00 20 53 4E 47
7FAB:  00 20 44 42 4C 00 20 20 20 44 49 4D 3A 00 0D 4F
7FBB:  55 54 20 4F 46 20 53 54 52 49 4E 47 20 53 50 41
7FCB:  43 45 0D 00 0D 41 52 52 41 59 3A 20 00 00 42 4C
7FDB:  45 27 F0 00 00 00 00 00 0D 4E 4F 54 20 46 4F 55
7FEB:  4E 44 0D 00 0D 45 4D 50 54 59 20 54 41 42 4C 45
7FFB:  0D 00 45 46
```

JUNIOR MATHS VZ 200

```
6 CLEAR1000:CLS:COLOR,1:REM COPYRIGHT - R. CARSON - 1983.
7 FORPO=0TO223 STEP1:PRINT@PO,CHR$(160);:NEXT
10 COLOR 6
20 PRINT@67,"                                "
30 PRINT@99,"                                "
40 PRINT@131,"                                "
41 PRINT@163,"                                ":SOUND30,2
42 PRINT@256,"                                "
45 FORL=1TO800:NEXT
46 A$="       WRITTEN BY R. CARSON       "
47 FORL=1TOLEN(A$)
48 PRINT@256,RIGHT$(A$,L);:NEXT
49 FORT=1TO2500:NEXT
50 T$="       WRITTEN BY R. CARSON       "
51 FORP=LEN(T$)TO1STEP-1:PRINT@256,RIGHT$(T$,P);:NEXT
57 B$="   ENJOY THIS EDUCATIONAL GAME  "
58 FORL=1TOLEN(B$)
59 PRINT@256,RIGHT$(B$,L);:NEXT
60 FORJ=1TO2500:NEXT
61 T$="   ENJOY THIS EDUCATIONAL GAME  "
62 FORL=LEN(T$)TO1STEP-1:PRINT@256,RIGHT$(T$,L);:NEXT
63 FORI=1TO800:NEXTI
70 SOUND20,3:PRINT"     YOUR CHOICE OF PROBLEMS"
71 PRINT:PRINT"          A = ADDITION"
72 PRINT"          D = DIVISION"
73 PRINT"          S = SUBTRACTION"
74 PRINT"          M = MULTIPLICATION"
79 K$=INKEY$
80 A$=INKEY$:IFA$=""THEN80
81 IFA$="M"GOTO60662
82 IFA$="D"GOTO60665
83 IFA$="A"GOTO60669
84 IFA$="S"GOTO60672
85 IFA$<>"M"ANDA$<>"D"ANDA$<>"A"ANDA$<>"S"THEN76
89 REM
100 C=0:G=0:P=0
101 CLS:COLOR,0
110 COLOR7:PRINT@32,"                                "
120 COLOR7:PRINT@64,"                                "
125 COLOR 2
130 PRINT@97,"                                ":COLOR7:PRINT@110,"                                "
132 COLOR 2
135 PRINT@129,"                                ":COLOR7:PRINT@142,"                                "
140 COLOR2
145 PRINT@161,"                                ":COLOR7:PRINT@174,"                                "
147 COLOR2
150 PRINT@193,"                                ":COLOR7:PRINT@206,"                                "
155 COLOR 2
160 PRINT@225,"                                ":COLOR7:PRINT@238,"                                "
165 COLOR 2
170 PRINT@257,"                                ":COLOR7:PRINT@270,"                                "
175 COLOR2
180 PRINT@289,"                                ":COLOR7:PRINT@302,"                                "
185 COLOR2
190 PRINT@321,"                                ":COLOR7:PRINT@334,"                                "
195 COLOR 4
200 PRINT@353,"                                ":COLOR7:PRINT@366,"                                "
203 COLOR7:PRINT@398,"                                "
205 COLOR 3
207 PRINT@385,"                                "
210 PRINT@417,"                                ":COLOR7:PRINT@430,"                                "
215 COLOR 3
220 PRINT@449,"                                ":COLOR7:PRINT@462,"                                "
223 SOUND30,5
225 COLOR,0
```

```
228 IFA$="D"THENPRINT@83,"DIVISION":SOUND30,2
229 IFA$="A"THENPRINT@83,"ADDITION":SOUND30,2
230 IFA$="S"THENPRINT@81,"SUBTRACTION":SOUND30,2
231 IFA$="M"THENPRINT@79,"MULTIPLICATION":SOUND30,2
250 PRINT@208,"I WILL ASK YOU":SOUND30,3
252 COLOR2:PRINT@129," ▄▄▄ ▄▄▄▄ ▄▄ "
255 PRINT@240,"SOME PROBLEMS,":SOUND30,3
256 COLOR2:PRINT@129," ▄▄▄ ▄▄▄▄ ▄▄ "
257 PRINT@272,"IF YOU GET";QW;:SOUND30,3
258 COLOR2:PRINT@129," ▄▄▄ ▄▄▄▄ ▄▄ "
260 PRINT@304,"CORRECT, THE  ":SOUND30,3
262 COLOR2:PRINT@129," ▄▄▄ ▄▄▄▄ ▄▄ "
265 PRINT@336,"WATER WILL GET":SOUND30,3
266 COLOR2:PRINT@129," ▄▄▄ ▄▄▄▄ ▄▄ "
267 PRINT@368,"DEEPER.       ":SOUND30,3
268 COLOR2:PRINT@129," ▄▄▄ ▄▄▄▄ ▄▄ "
270 FORI=1TO5000:NEXTI
273 COLOR7
274 PRINT @79,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
275 PRINT@176,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
276 PRINT@208,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
277 PRINT@240,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
278 PRINT@272,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
279 PRINT@304,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
280 PRINT@336,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
290 PRINT@368,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
60020 COLOR,0
60022 IFA$="M"GOSUB60700
60030 IFA$="D"GOSUB60710
60035 IFA$="A"GOSUB60720
60040 IFA$="S"GOSUB60730
60045 COLOR 2:PRINT@257," ▄▄▄▄▄▄▄▄▄▄ "
60050 IFA$="M"THENA=Y*Z:PRINT@176,Y"X"Z"="
60055 IFA$="D"THENB=Y*Z:A=Y:PRINT@176,B"÷"Z"="
60060 IFA$="A"THENA=V+W:PRINT@176,V"+"W"="
60065 IFA$="S"THENJ=V+W:A=V:PRINT@176,J"-"W"="
60110 PRINT@240,"ANSWER";:INPUTD$
60115 FORZ=1TOLEN(D$)
60119 NEXTZ
60120 X=VAL(D$)
60125 IFX=ATHENC=C+1:SOUND25,2:PRINT@82,C;"CORRECT"
60130 COLOR 2:PRINT@257," ▄▄▄▄▄▄▄▄▄▄ "
60136 IFC=QWTHEN60261
60140 IFX<>ATHEN60175
60150 FORT=1TO1000
60155 NEXTT
60160 COLOR7:PRINT@175,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
60165 COLOR7:PRINT@304,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄":PRINT@239,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
60170 COLOR7:PRINT@240,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄":GOTO60020
60175 COLOR2:PRINT@257," ▄▄▄▄▄▄▄▄▄▄ "
60180 SOUND 16,3
60190 SOUND 11,2
60200 SOUND 11,1
60210 SOUND 13,3
60220 SOUND 11,3
60230 SOUND 0,2
60240 SOUND 15,4
60250 SOUND 16,4
60251 PRINT@368,"ANSWER IS"A:G=G+1:FORV=1TO2500:NEXT
60253 COLOR7:PRINT@175,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
60255 COLOR7:PRINT@335,"▄▄▄▄▄▄"
60256 COLOR7:PRINT@367,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
60257 COLOR7:PRINT@240,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
60260 GOTO60020
60261 FORI=1TO1500:NEXTI:COLOR7:SOUND20,3
60262 PRINT @82,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
60263 PRINT@176,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
60264 PRINT@208,"▄▄▄▄▄▄I▄▄▄▄▄▄"
60265 PRINT@240,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
60266 PRINT@272,"▄▄▄▄▄DID▄▄▄▄▄▄"
60267 PRINT@304,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
60268 PRINT@336,"▄▄▄PROMISE▄▄▄▄"
60269 PRINT@368,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄":FORI=1TO1500:NEXTI
60270 SOUND20,2:COLOR3:PRINT@358,"▄▄":FORI=1TO200:NEXTI
60271 COLOR3:PRINT@357,"▄▄▄▄":FORI=1TO200:NEXTI
60272 COLOR3:PRINT@356,"▄▄▄▄▄":FORI=1TO200:NEXTI
60273 COLOR3:PRINT@355,"▄▄▄▄▄▄":FORI=1TO200:NEXTI
60274 COLOR3:PRINT@354,"▄▄▄▄▄▄▄":FORI=1TO200:NEXTI
60275 COLOR2:PRINT@257," ▄▄▄▄▄▄▄▄ "
60276 COLOR3:PRINT@353,"▄▄▄▄▄▄▄▄▄▄":FORI=1TO200:NEXTI
60278 COLOR3:PRINT@321,"▄▄▄▄▄▄▄▄▄▄":FORI=1TO200:NEXTI
60280 COLOR3:PRINT@289,"▄▄▄▄▄▄▄▄▄▄":FORI=1TO200:NEXTI
60290 COLOR3:PRINT@257,"▄▄▄▄▄▄▄▄▄▄":FORI=1TO200:NEXTI
60300 COLOR3:PRINT@225,"▄▄▄▄▄▄▄▄▄▄":FORI=1TO200:NEXTI
60305 COLOR2:PRINT@129," ▄▄▄▄▄▄▄▄▄▄ "
60310 COLOR3:PRINT@193,"▄▄▄▄▄▄▄▄▄▄":FORI=1TO200:NEXTI
60320 COLOR3:PRINT@161,"▄▄▄▄▄▄▄▄▄▄":FORI=1TO200:NEXTI
60330 COLOR3:PRINT@129,"▄▄▄▄▄▄▄▄▄▄":FORI=1TO200:NEXTI
60340 COLOR3:PRINT@97, "▄▄▄▄▄▄▄▄▄▄"
60350 FORI=1TO1500:NEXTI
60351 QW=0
60352 SOUND20,3:PRINT@210,C;"CORRECT"
60353 PRINT@274,G;"WRONG"
60354 PRINT@338,INT(C*100/(C+G));"PERCENT"
60355 FORI=1TO1500:NEXTI
60356 COLOR7:PRINT@337,"▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
60359 COLOR 7
60360 SOUND30,2:PRINT@208,"▄▄▄▄ ENTER  Y ▄▄▄▄"
60370 PRINT@271,"▄ TO PLAY AGAIN"
60380 PRINT@334,"▄ N TO FINISH ";:INPUTC$
60390 IFC$="Y"THEN 60750
60400 CLS:PRINT:PRINT:PRINT:PRINT"                    BYE":END
60662 Y=RND(12):Z=RND(12)
60663 QW=RND(50):IFQW<10THENQW=10
60664 GOTO100
60665 Y=RND(12):Z=RND(12)
60666 QW=RND(50):IFQW<10THENQW=10
60668 GOTO100
60669 V=RND(100):W=RND(100)
60670 QW=RND(50):IFQW<10THENQW=10
60671 GOTO100
60672 V=RND(100):W=RND(100)
60673 QW=RND(50):IFQW<10THENQW=10
60675 GOTO100
60700 Y=RND(12):Z=RND(12):RETURN
60710 Y=RND(12):Z=RND(12):RETURN
60720 V=RND(100):W=RND(100):RETURN
60730 V=RND(100):W=RND(100):RETURN
60750 CLS:PRINT:PRINT:PRINT:GOTO70
```

BATTLESHIPS VZ 200

```
3 CLS:COLOR,1:PRINT@170,"FOR VZ-200"
4 PRINT@201,"BY R. CARSON":PRINT@235,"ADELAIDE"
5 PRINT@33,"***THE GAME OF BATTLESHIPS***":REM COPYRIGHT
6 PRINT@425,"INSTRUCTIONS?":PRINT@456,">>Y=YES  N=NO<<"
7 K$=INKEY$
8 I$=INKEY$:IF I$=""THEN8
9 IF I$="Y"THEN 12
10 IF I$="N"THEN90
11 IF I$<>"Y"THEN7:IF I$<>"N"THEN7
12 CLS:PRINT"THE PLAYING AREA REPRESENTS AN  ";
13 PRINT"AREA OF SEA. THE COMPUTER IS    ";
14 PRINT"CONTROLLING TEN SHIPS, A BATTLE-";
15 PRINT"SHIP, 2 CRUISERS, 3 DESTROYERS  ";
16 PRINT"AND 4 SUBMARINES. OF COURSE, I  ";
17 PRINT"CAN'T TELL YOU WHERE THEY ARE.  ";
18 PRINT"ONLY THE COMPUTER KNOWS, UNTIL  ";
19 PRINT"YOU HIT THEM. THE SHIPS ARE     ";
20 PRINT"DIFFERENT SIZES, AND ARE IDENTI-";
21 PRINT"FIED BY THE INITIAL LETTER. THE ";
22 PRINT"BATTLESHIP OCCUPIES FOUR SQUARES";
23 PRINT"LIKE THIS: BBBB, ACROSS OR DOWN."
24 PRINT:PRINT"   PRESS <SPACE> TO CONTINUE"
25 K$=INKEY$
26 I$=INKEY$:IF I$<>" "THEN 26
32 CLS:PRINT"THE CRUISERS THREE SQUARES, THE ";
33 PRINT"DESTROYERS TWO SQUARES, AND THE ";
34 PRINT"SUBMARINES ONE SQUARE, ALWAYS IN";
35 PRINT"A STRAIGHT LINE. SHIPS MAY TOUCH";
36 PRINT"OR LAY ALONGSIDE EACH OTHER.YOU ";
37 PRINT"FIRE A SHOT BY GIVING TWO       ";
38 PRINT"NUMBERS. THE FIRST ON THE LEFT, ";
39 PRINT"THE SECOND AT THE TOP. IF YOU   ";
40 PRINT"HIT ANYTHING, A LETTER WILL BE  ";
41 PRINT"PRINTED TO TELL YOU WHICH TYPE  ";
42 PRINT"OF SHIP YOU HIT. TO SINK IT, YOU";
43 PRINT"MUST HIT ALL THE SQUARES OF     THAT ";
44 PRINT"PARTICULAR SHIP."
45 PRINT:PRINT"   PRESS <SPACE> TO CONTINUE"
46 K$=INKEY$
47 I$=INKEY$:IF I$<>" "THEN47
52 CLS:PRINT"IF YOU MISS, THEN * IS PRINTED  ";
53 PRINT"TO REMIND YOU THAT YOU HAVE SHOT";
54 PRINT"INTO THAT SQUARE BEFORE."
55 PRINT:PRINT"YOUR NUMBER OF SHOTS IS SHOWN AT";
56 PRINT"THE BOTTOM OF THE SCREEN AND THE";
57 PRINT"BEST SCORE YOU ACHIEVED DURING A";
58 PRINT"SERIES OF GAMES. THE GAME ENDS  ";
59 PRINT"WHEN ALL SHIPS HAVE BEEN SUNK."
60 PRINT:PRINT:PRINT"          HAPPY HUNTING"
61 PRINT:PRINT:PRINT"     PRESS <SPACE> TO START"
62 K$=INKEY$
63 I$=INKEY$:IF I$<>" "THEN 63
90 CLS
95 X=0
100 A=100
110 DIM G(100)
120 D=0
125 CLS:PRINT@196,"WAIT---ARRANGING FLEET"
130 C=0
140 FOR B=1 TO 100
150 G(B)=0
160 NEXT B
170 E=4
180 F=1
190 H=INT(RND(X)*2)
195 W=0
200 IF H=0 THEN J=RND(9)
202 IF H=1 THEN J=RND(4)
205 IF H=1 THEN K=RND(9)
212 IF H=0 THEN K=RND(4)
220 L=0
230 P=10*J+K
250 FOR M=0 TO (E-1)
255 IF H=0 THEN R=P+M
260 IF H=1 THEN R=P+10*M
280 IF L=0 AND G(R)<>0 THEN W=W+1
290 IF L=1 THEN G(R)=E
300 NEXT M
305 IF W>0 AND W<10 THEN 190
306 IF W=10 THEN 140
310 IF L=1 THEN 400
320 L=1
330 GOTO 250
400 F=F+1
410 IF F<4 THEN E=3
420 IF F>3 AND F<7 THEN E=2
430 IF F>6 THEN E=1
440 IF F=11 THEN 700
445 GOTO 190
450 PRINT@435,"    ";:INPUTS
458 IF S<11 THEN 450
460 IF S>99 THEN 450
465 T=INT((S)/10)
470 U=S-T*10
472 IF G(S)=5 THEN 450
475 IF G(S)=4 THEN S1$="B"
480 IF G(S)=3 THEN S1$="C"
485 IF G(S)=2 THEN S1$="D"
490 IF G(S)=1 THEN S1$="S"
495 IF G(S)=0 THEN S1$="*"
500 V=U*2+T*32+101
510 PRINT@V,S1$
520 C=C+1
530 IF G(S)>0 THEN D=D+1
535 G(S)=5
540 PRINT@418,"SHOTS:";C;
550 IF D<20 THEN 450
560 IF A<82 THEN PRINT@457,"BEST SCORE:";A
570 IF C<A THEN A=C
580 PRINT@482,"ANOTHER GAME?>>Y=YES  N=NO<<";
585 K$=INKEY$
590 I$=INKEY$:IF I$=""THEN590
595 IF I$="Y"THEN120
597 IF I$="N"THENCLS:END
600 IF I$<>"Y"THEN 585
610 IF I$=<>"N"THEN 585
700 CLS:PRINT@39,"***BATTLESHIPS***"
720 PRINT:PRINTTAB(7)"1 2 3 4 5 6 7 8 9"
730 FOR N=1 TO 9
740 PRINTTAB(4);N;". . . . . ."
750 NEXT N
760 GOTO 450
```

NEXT MONTH'S ISSUE

Next month's issue will contain at least the following programs plus the usual features and articles. (80) after a program title indicates that the program will be for TRS-80 Model 1/3 or System 80/Video Genie. A (CC) indicates that the program will be for the TRS-80 Colour Computer and (VZ) that the program is for the VZ-200.

DOG RACE — VZ

This program was published in MICRO-80 some time ago for '80 computers. Here is the opportunity for VZ owners to gamble their all on their favourite dog.

CONTEST LOG — VZ

Many of our readers are Amateur Radio enthusiasts. This program was designed to assist in RD Contests but is useful for many other type of log for which you wish to record a hard copy of call signs worked.

TOUCH TYPING — '80

This program will assist you to improve your keyboard skills. All the super-dooper programmers aids are of little benefit unless you can touch type. A few hours spent at the keyboard with this program will save you many hours later.

TRACK 80 — '80

Here is an arcade type car racing game guaranteed to test your reflexes and ability to make quick decisions.

SORT UTILITY — '80

This is a short Bubble Sort routine which you can use in your own programs. It is able to sort 100 integers before you get your finger off the Enter key.

LUNAR LANDER — COCO

Try to land your Lunar Module in one of three deep craters and gain points for successes.

APPLICATION FOR PUBLICATION OF A PROGRAM IN MICRO-80

Date .....

To **MICRO-80**
SOFTWARE DEPT.,
P.O. BOX 213,
GOODWOOD, S.A. 5034

Please consider the enclosed program for publication in MICRO-80.

Name .....

Adress .....

..... Postcode .....

*** CHECK LIST ***

Please ensure that the cassette or disk is clearly marked with your name and address, program name(s), Memory size, Level I, II, System 1 or 2, Edtasm, System, etc. The use of REM statements with your name and address is suggested, in case the program becomes separated from the accompanying literature.

Ensure that you supply adequate instructions, notes on what the program does and how it does it, etc.

For system tapes, the start, end, and entry points, etc.

The changes or improvements that you think may improve it.

Please package securely — padabags are suggested — and enclose stamps or postage if you want your cassette or disk returned.

CASSETTE/DISK EDITION INDEX

The cassette edition of MICRO-80 contains all the applicable software listed each month, on cassette. For machine language programs copies of both the source and object file are provided. All programs are recorded twice. Level 1 programs can only be loaded into a Level 2 machine if the 'Level 1 in Level 2' program from the MICRO-80 Software Library — Vol. 1 is loaded first.

***Note:** System 80/Video Genie computers have had different tape-counters fitted at different times. The approximate start positions shown are correct for the very early System 80 without the volume control or level meter. They are probably incorrect for later machines. The rates for a cassette subscription are printed on the inside front cover of each issue of the magazine.*

The disk edition contains all applicable programs which can be executed from disk. Level 1 disk programs are saved in NEWDOS format. Users require the Level 1/CMD utility supplied with NEWDOS+ or NEWDOS 80 version 1.0 to run them.

VZ200 programs are not currently available on cassette or disk.

				Approx. Start Position		
Side 1	**Type**	**I.D.**	**Disk Filespec**	**CTR-41**	**CTR-80**	**System 80**
Sharemarket	LII/16K	S	SHAREMAR/BAS	10	6	4
Words and Meanings	LII/16K	W	WORDS/BAS	150	85	57
Sirius Adventure	32K/Disk	A	SIRIUS/BAS	180	102	68
Disk Directory Recorder	32K/Disk/Mod III	3	DIRECT32/M3	255	144	96
Disk Directory Recorder	48K/Disk/Mod III	4	DIRECT48/M3	310	175	117
Array Utility (Address 7BAB 7FFE 0000)	LII/16K	ARAY16	ARRAY16K/CMD	365	206	138
Array Utility (Addresses BBAB BFFE 0000)	LII/32K	ARAY32	ARRAY32K/CMD	380	214	144
Array Utility (Addresses FBAB FFEE 0000)	LII/48K	ARAY48	ARRAY48K/CMD	395	223	149
Array Demo	LII/16K	A	AARRAY/BAS	410	231	155
Dogfight	COCO	DOGFIGHT	—	430	243	163
Side 2						
Dogfight	COCO	DOGFIGHT	—	10	6	4
Array Utility	EDTASM	AARRAY	AARRAY/EDT	40	23	15
Sharemarket	LII/16K	S	SHAREMAR/BAS	150	85	57
Words and Meanings	LII/16K	W	WORDS/BAS	270	152	102
Sirius Adventure	32K/Disk	S	SIRIUS/BAS	295	166	112
Disk Directory Recorder	48K/Disk/Mod III	4	DIRECT48/M3	360	203	136
Array Utility	LII/16K	ARAY16	ARRAY16K/CMD	410	231	155
Array Demo	LII/16K	A	AARRAY/BAS	425	240	161

TO:
MICRO-80, P.O. BOX 213, GOODWOOD,
SOUTH AUSTRALIA. 5034.

Please RUSH to me the items shown below:

$ enclosed Date .....

..... 12 month subscription to MICRO-80
..... 12 month subs. to MICRO-80, plus the cassette edition
..... 12 month subs. to MICRO-80, plus the disc edition
..... The latest issue of MICRO-80 (see inside front cover for prices)

FOR TRS-80 ☐ 1 ☐ 2/16 ☐ 3 — KRAM ☐ TAPE ☐ DISK
SYSTEM 80 MARK ☐ 1 ☐ 11 — KRAM

DESCRIPTION	QTY	PRICE

TOTAL ENCLOSED WITH ORDER P/H

☐ Cheque ☐ Bankcard ☐ Money Order Total

Bankcard Account Number

Signature ..... Exp. End .....

NAME .....

ADDRESS ..... Postcode .....

*Post/Handling charge on all Software ordered — $4.00.

MOLYMERX

Australia's broadest range of software for TRS-80's and SYSTEM 80's

MOLYMERX has the Australian distribution rights for literally hundreds of top grade programs from American, Canadian and British publishers. From games to utilities, from DOS's to Databases, if it's top quality then MOLYMERX almost certainly has it.

Now, MOLYMERX is being distributed in Australia by MICRO-80. To help you chose from the incredibly wide range of programs available, you may purchase a MOLYMERX catalogue. For only $5.00 you receive over 80 pages of what is virtually an encyclopedia of '80 software plus regular updates for 12 months. The useful information contained in this catalogue is worth many times its cost.

There are now generous BULK BUYING DISCOUNTS of 10% off list price for single orders in excess of $500 or 15% for single orders in excess of $1,000. So get together with your friends or User Group members to place a combined order and save yourselves real $$$.

EXPANSION INTERFACES FOR SYSTEM 80 and TRS-80 COMPUTERS

MICRO-80's new family of expansion interfaces for the System 80 and TRS-80 offer unprecendented features and reliability including:

Up to 32K STATIC RAM : to ensure high noise immunity and reliability

Centronics Printer Port: The Systems 80 Expansion Interface has a double-decoded port to respond to both port FD and memory address 37E8H, thus overcoming one of the major incompatabilities with the TRS-80.

RS232 Communications Port: for communicating via modem or direct link to other computers

Single Density Disk Controller: for complete compatability with all Disk Operating Systems

Supports double-sided Disk Drives up to 80 tracks: with a suitable disk operating system such as DOSPLUS, NEWDOS 80 or LDOS, the interface will support single or double sided drives of 35-80 track capacity.

Economical double density: an economical, high quality double-density upgrade will be released shortly to enable you to increase the capacity of your disk drives by 80%.

Real time clock interrupt: provides software clock facility used by most DOS's.

SYSTEM-80 EXPANSION IN/FACE		TRS-80 EXPANSION INTERFACE	
WITH ØK RAM	$450.00	WITH ØK RAM	$450.00
ADDITIONAL 16K RAM	99.00	ADDITIONAL 16K RAM	99.00
ADDITIONAL 32K RAM	198.00	ADDITIONAL 32K RAM	198.00

SYSTEM 80 AND TRS-80 PRINTER INTERFACES $99 + $3.00 p&p

For those who wish to add a printer to their SYSTEM 80. MICRO-80's new printer interface provides the ideal solution. Double-decoded to both port FD and address 37E8H, this interface overcomes one of the major incompatabilities between the SYSTEM 80 and the TRS-80. Price includes a Centronics printer cable. Operates with Centronics compatible printers including GP-80 and GP-100.

MICRO-80

LEVEL 2 ROM

ASSEMBLY LANGUAGE TOOLKIT

by Edwin Paay

FOR TRS-80 MODEL 1, MODEL 3 AND SYSTEM 80/VIDEO GENIE

This is a new package consisting of two invaluable components:

- **A ROM REFERENCE** Manual which catalogues, describes and cross-references the useful and usable ROM routines which you can incorporate into your own machine language or BASIC programs.
- **DBUG**, a machine language disassembling debugging program to speed up the development of your own machine language programs. DBUG is distributed on a cassette and may used from disk or cassette.

Part 1 of the ROM REFERENCE manual gives detailed explanations of the processes used for arithmetical calculations, logical operations, data movements etc. It also describes the various formats used for BASIC, System and Editor/Assembly tapes. There is a special section devoted to those additional routines in the TRS-80 Model 3 ROM. This is the first time this information has been made available, anywhere. Differences between the System 80/Video Genie are also described. Part 1 is organised into subject specific tables so that you can quickly locate all the routines to carry out a given function and then choose the one which meets your requirements.

Part 2 gives detailed information about each of the routines in the order in which they appear in the ROM. It describes their functions, explains how to use them in your own machine language programs and notes the effect of each on the various Z80 registers.

Part 2 also details the contents of system RAM and shows you how to intercept BASIC routines. With this knowledge, you can add your own commands to BASIC, for instance, or position BASIC programs in high memory — the only restriction is your own imagination!

The Appendices contain sample programmes which show you how you can use the ROM routines to speed up your machine language programs and reduce the amount of code you need to write.

DBUG: Eddy Paay was not satisfied with any of the commercially available debugging programs, so he developed his own. DBUG: allows you to single-step through your program; has a disassembler which disassembles the next instruction before executing it or allows you to bypass execution and pass on through the program, disassembling as you go; displays/edits memory in Hex or ASCII; allows Register editing; has the ability to read and write System tapes and all this on the bottom 3 lines of your screen, thus freeing the rest of the screen for program displays. Four versions of DBUG are included in the package to cope with different memory sizes.

The best news of all is the price. The complete Level 2 ROM ASSEMBLY LANGUAGE TOOLKIT is only:

— Aus. $29.95 + $2.00 p&p
— UK £18.00 + £1.00 p&p

SPECIAL OFFER TO OWNERS OF THE LEVEL II ROM REFERENCE MANUAL ...
UPGRADE TO THIS ASSEMBLY LANGUAGE TOOKIT FOR ONLY $19.95!
Send back your original Level II ROM Reference Manual plus a cheque, money order or Bankcard authorisation for $19.95 plus $2.00 p&p and we will send you the new ASSEMBLY LANGUAGE TOOLKIT

MICRO-80